

Swarm Documentation



ⓘ Finding versions

For the *live and current* online version, see [DataCore Swarm Documentation](#).

To view links to PDFs for all products and releases, see the [Documentation Archive](#).

- [How to Upgrade Swarm](#)
- [Swarm Deployment](#)
 - [Swarm 14.0 VMware Bundle Package](#)
 - [Use Cases and Architectures](#)
 - [Content UI Installation](#)
 - [Swarm Storage UI Installation](#)
 - [Deployment Planning](#)
 - [SwarmFS Implementation](#)
 - [Migrating from Traditional Storage](#)
 - [Elasticsearch Implementation](#)
 - [Content Gateway Implementation](#)
 - [Deployment Process](#)
 - [Network Infrastructure](#)
 - [Hardware Setup](#)
 - [Storage Implementation](#)
 - [SCS Implementation](#)
 - [SCS 2.0 Installation](#)
- [Swarm Administration](#)
 - [Swarm Storage UI](#)
 - [Elasticsearch for Swarm](#)
 - [Swarm Storage Cluster](#)
 - [Swarm Cluster Services \(SCS\)](#)
 - [SCS Administration](#)
 - [Swarm Content Gateway](#)
 - [Swarm Content UI](#)
 - [Bucket Lifecycle Policy](#)
- [Documentation Archive](#)
- [Swarm Release Notes](#)
 - [Swarm 14 Highlights](#)
 - [Content Gateway Release Notes](#)

- [Content UI Release Notes](#)
- [SDK Release Notes](#)
- [SwarmFS Release Notes](#)
- [Swarm Storage Release Notes](#)
- [Storage UI Release Notes](#)
- [Swarm Platform Release Notes](#)
- [Swarm Development](#)
 - [S3 Protocol Interface](#)
 - [Storage SCSP Development](#)
 - [SCS CLI Commands](#)
 - [Content Application Development](#)
 - [Swarm SDK](#)

How to Upgrade Swarm

- [Get Products and Docs](#)
- [Upgrade Planning](#)
- [Upgrade Paths](#)
- [Upgrading from Unsupported Elasticsearch](#)

Get Products and Docs

1. Navigate to the [Downloads section](#) on the [DataCore Support Portal](#).
2. Scroll down and open the Swarm toggle
 - These bundles are updated and the ZIP file name changes to reflect the *release date for the bundle* when an updated component version is available such as an updated release of Content Gateway.
3. Download the corresponding PDF from the [Documentation Archive](#) page.
 - The comprehensive PDF matching all components at those versions is uploaded to the [Documentation Archive](#) using a filename including the same *release date for the matching bundle*.
 - The Swarm [online documentation](#) is continually updated to the current release; because the PDF is a snapshot tied to a bundle release, consider it the definitive source for *that* bundle.
 - The DataCore Support team's [searchable Knowledge Base](#) contains technical articles and the latest documentation, which may be *newer* than the installed version.
4. Expand the software bundle. In the top-level directory of the bundle, locate and read the `README.txt` for version guidance on using the bundle.
5. Open the PDF for the bundle and see the [Release Notes for each component](#), which include upgrade instructions as well as changes and watch items.

Upgrade Planning

1. **Plan upgrade impacts** – Review and plan for [this release's upgrade impacts](#) and the impacts for each of the releases since the currently running version. For Swarm 9 impacts, see [Swarm Storage 9 Releases](#).
2. **No volume retires** – Do not start any elective volume retirements during the upgrade. Wait until the upgrade is complete before initiating any retires, or verify they are complete before upgrading.
3. **Choose the reboot type** – Swarm supports rolling upgrades (a single cluster running mixed versions during the upgrade process) and requires no data conversion unless noted for a release. Upgrades are performed without scheduling an outage or bringing down the cluster. Restart the swarm nodes one at a time with the new version and the cluster continues serving applications during the upgrade process.
 - **Rolling upgrade:** Reboot one node at a time and wait for its status to show as "OK" in the UI before rebooting the next node.
 - **Alternative:** Reboot the entire cluster at once after the software on all USB flash drives or the centralized configuration location has been updated.
4. **Follow the Upgrade Path**, below.
5. Review the [Application and Configuration Guidance](#).

Upgrade Paths

Swarm upgrade paths depend on the implementation environment. See [Installing and Initializing Swarm Storage](#) for first time Swarm installations.

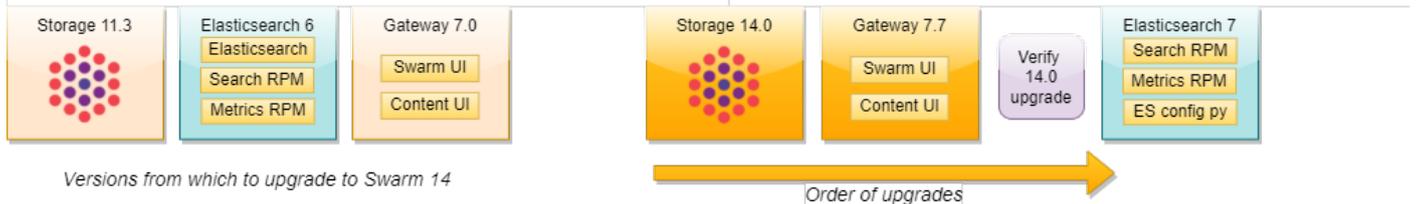
Component-only upgrades

Not all components update in every release; some upgrades contain a single RPM. The currently running Swarm version determines which components require an upgrade. Follow the release tables on the [Documentation Archive](#) to track component versions.

Upgrade Path options:

1. Running Elasticsearch 2.3.3 or 5.6.12, see [Upgrading from Unsupported Elasticsearch](#), below.
2. Not using a Search feed or running Elasticsearch 6.8.6, continue with the upgrade path.

CSN	No CSN
<ol style="list-style-type: none"> 1. Run settings checker and review upgrade impacts 2. Review and address report with Support 3. Download the CSN Swarm bundle from the Downloads section on the DataCore Support Portal 4. Upgrade the Storage RPM or run the script (see flowchart) 5. Select the new Storage version and reboot the cluster to activate it 6. Standalone Gateway: <ol style="list-style-type: none"> a. Upgrade Gateway (with Swarm UI, Content UI) 7. Standalone Elasticsearch: <ol style="list-style-type: none"> a. Verify Swarm 14 is working (no downgrading after ES goes to 7) b. On ES 6, first install the Search and Metrics RPMs. c. Run the config script provided, which installs and configures ES 7: <code>configure_elasticsearch_with_swarm_search.py</code> 	<ol style="list-style-type: none"> 1. Run settings checker and review upgrade impacts 2. Review and address report with Support 3. Download the Swarm bundle from the Downloads section on the DataCore Support Portal 4. Upgrade Storage (fsimage/kernel files via USB key or PXE server) 5. Complete required cluster reboot 6. Standalone Gateway: <ol style="list-style-type: none"> a. Upgrade Gateway (with Swarm UI, Content UI) 7. Standalone Elasticsearch: <ol style="list-style-type: none"> a. Verify Swarm 14 is working (no downgrading after ES goes to 7) b. On ES 6, first install the Search and Metrics RPMs. c. Run the config script provided, which installs and configures ES 7: <code>configure_elasticsearch_with_swarm_search.py</code>



Upgrading from Unsupported Elasticsearch

Contact DataCore Support to guarantee a smooth migration process with no down-time if running unsupported Elasticsearch versions 2.3.3 or 5.6.12.

Swarm 9.6	Gateway 5.4	Elasticsearch 2.3.3	<i>Migration to ES 6.8.6 required (new cluster and new Search feed)</i>
Swarm 11.3	Gateway 5.4.1	Elasticsearch 2.3.3	
	Gateway 7.0	Elasticsearch 5.6.12	
Swarm 14.0	Gateway 7.7	Elasticsearch 6.8.6	<i>Upgrade in-place to ES 7.5.2</i>
		Elasticsearch 7.5.2	<i>Upgrade in-place to future ES version</i>

Swarm 14.0 Update 2	Gateway 7.8	Elasticsearch 7.5.2	<i>Upgrade in-place to future ES version</i>
------------------------	-------------	---------------------	--

The high-level upgrade sequence is as follows:

1. **Swarm 11** bundle:

- a. Upgrade Swarm to 11.3, as guided by DataCore Support and the [Settings Checker](#) report.
- b. Upgrade to Gateway 5.4.1 if currently running Elasticsearch 2.3.3. Upgrade to Gateway 7.0 if currently running Elasticsearch 5.6.12. Refer to [Upgrading from Gateway 5.x](#).
- c. Add an Elasticsearch 6 cluster and start a search feed, leaving the old feed as primary (see [Migrating from Older Elasticsearch](#)).
- d. When the feed completes, make it primary.
- e. Upgrade to Gateway 7.0 if currently running Gateway 5.4.1.
- f. Configure gateway.cfg `indexerHosts` to point to the new Elasticsearch 6 cluster and restart CloudGateway, using `sudo systemctl restart cloudgateway` command.

2. **Swarm 14** bundle:

- *Follow the appropriate column in **Upgrade Paths**, above.*
This is the general order:
 - a. Upgrade to Swarm 14.
 - b. Upgrade to Gateway 7.7 (from version 5.4.1, follow [Upgrading from Gateway 5.x](#))
 - c. Verify Swarm operations (this is the time to downgrade).
 - d. Run the ES config script to [upgrade in-place](#) to Elasticsearch 7.
 - e. Upgrade Gateway 7.7 to Gateway 7.8.

Swarm Deployment

Swarm combines the scalable software-defined object storage of Swarm Storage with components that support many types of implementations.

Install the components in this order to implement Swarm:

SCS	A single package for configuring Swarm storage	Install , Setup , Run , Configure
Storage Cluster	Cluster for Swarm storage nodes	Requirements , Network , Install , Configure
Elasticsearch	Cluster for search	Requirements , Prepare , Install , Configure
Content Gateway	Gateway for cloud-based client access (S3)	Requirements , Install , Configure
Storage UI	Website for storage cluster management	Install SCS
Content UI	Website for cloud content management	Install
SwarmFS	Optional connector for NFS clients	Install , Configure

Before installing any Swarm packages, complete the planning and preparation of the Swarm environment.

- [Swarm 14.0 VMware Bundle Package](#)
- [Use Cases and Architectures](#)
- [Content UI Installation](#)
- [Swarm Storage UI Installation](#)
- [Deployment Planning](#)
- [SwarmFS Implementation](#)
- [Migrating from Traditional Storage](#)
- [Elasticsearch Implementation](#)
- [Content Gateway Implementation](#)
- [Deployment Process](#)
- [Network Infrastructure](#)
- [Hardware Setup](#)
- [Storage Implementation](#)
- [SCS Implementation](#)
- [SCS 2.0 Installation](#)

Swarm 14.0 VMware Bundle Package

This bundle contains a collection of supported VMware templates for use in proof of concepts and production field deployments. Each VM contains all necessary DataCore Swarm software as well as dependencies. The software packages are pre-installed and unconfigured.

This package contains:

- SwarmTelemetry OVF file
- Corrected guest OS labels for all VMs
- VMware Hardware Level is set to 11 across all VMs
- SwarmTelemetry: DataCore branded email alert templates added

Prerequisites

- A dedicated vSwitch for the Swarm Storage Network before importing VMs with dual virtual network card.
- VMs created on VMware ESX 6.0 and must be loaded into 6.0U3 and above.
- If using vCenter, use the VMware vAPP OVF datacore-swarm-14.0.1-ESX.ovf to load the entire set of VMs in one command.
- If not using vCenter, use an individual OVF for each template. This requires specifying the associated disk for each VM.

Note:

- All VMs have dual virtual network cards; one for the public client network and another for the Swarm Storage Network.
- For SwarmSearch1, the public-facing network card is not connected for security reasons.
- VMware ESX 6.7U3 is required if using vCenter 6.7.
- All access credentials use the default password "datacore".

Disk to VM Mapping

The following is a sample mapping example of OVF with the associated disks:

OVF	Associated Disks
SwarmClusterServices.ovf	datacore-swarm-14.0.1-ESX-disk7.vmdk
SwarmContentGateway.ovf	datacore-swarm-14.0.1-ESX-disk6.vmdk
SwarmFS.ovf	datacore-swarm-14.0.1-ESX-disk5.vmdk
SwarmSearch1.ovf	datacore-swarm-14.0.1-ESX-disk3.vmdk
	datacore-swarm-14.0.1-ESX-disk4.vmdk
SwarmTelemetry.ovf	datacore-swarm-14.0.1-ESX-disk1.vmdk
	datacore-swarm-14.0.1-ESX-disk2.vmdk

VM Template Overview

The following is a sample example of the VM template:

VM	Hostname	vCPU	RAM	System Disk	Data Disk
SwarmClusterServices	swarmclusterservices	2	4 GB	50 GB	--
SwarmContentGateway	swarmcontentgateway	4	8 GB	50 GB	--
SwarmSearch	swarmsearch1	4	24 GB	30 GB	0.45 TB
SwarmNFS	swarmnfs	2	8 GB	16 GB	--
SwarmTelemetry	swarmtelemetry	1	1GB	20 GB	100 GB

Software Versions

The following software versions are required for Swarm 14.0 VMware bundle:

VM	Application	Version	Vendor
All	Centos	7.9	CentOS
SCS	Swarm Cluster Services	1.1.0	DataCore Software, Inc
GW	Content Gateway Web UI	7.5.0	DataCore Software, Inc
GW	Content Gateway	7.8.0	DataCore Software, Inc
SCS	Swarm	14.0.1	DataCore Software, Inc
GW	Storage Management UI	3.4.0	DataCore Software, Inc
NFS	Swarm NFS	3.2.0	DataCore Software, Inc
GW	Haproxy	1.8.17	https://www.haproxy.org/
ES	Elasticsearch	7.5.2	Elasticsearch
TM	Prometheus	2.32.1	https://prometheus.io/
TM	AlertManager	0.16.1	https://prometheus.io/
GW	Node Exporter	0.18.1	https://prometheus.io/
TM	Grafana	8.3.3	https://grafana.com/

References

Abbreviation	Full Form
SCS	Swarm Cluster Services
GW	Swarm Content Gateway
NFS	SwarmNFS
ES	SwarmSearch
TM	SwarmTelemetry

Use Cases and Architectures

Most use cases for Swarm involve ingesting petabytes of unstructured data, such as image, video, and document files, which must be secured, preserved, searched, and retrieved on demand.

- **Active Archive** – video evidence, medical imaging, cultural media
- **Cloud** – cloud services and hosting (multi-tenant), backup to the cloud
- **Content Delivery** – social media (millions of photos per day), streaming video (millions of videos), content publishing (millions of images)
- **Big Data** – evidence analysis, medical insurance records and analysis, IoT/M2M and analytics
- **Compliance** – legal documents, court materials, digital evidence

Swarm supports many usage scenarios based on four fundamental access methods:

Direct Access	Native (SCSP API)	<ul style="list-style-type: none"> • Native client/application integration using a vendor API (RESTful HTTP 1.1 compliant) • Expectation the application works directly with the object store
Web Access	Content Gateway (S3 API)	<ul style="list-style-type: none"> • Data in the object store is presented via web browser (Content UI) • S3 endpoint is provided • Support for actions such as upload, download, and browse • Back-end access to the object store are native API calls
File Protocol Gateway	SwarmFS (to Native SCSP)	<ul style="list-style-type: none"> • Provides translation of traditional file protocols (such as NFS and SMB) to object storage protocols • Usually translates to object store native API • Used as a “drop box” target for clients/applications coded to work with traditional filers • Advanced protocol gateway support for manipulation of metadata, in addition to data via traditional utilities (such as shell sessions) • Placement into object store supports alternate access methods (SCSP or S3) and metadata queries, listings, and collections
Automated Tiering	FileFly	<ul style="list-style-type: none"> • Application/agent integration (native API integration) • Agents exist on data sources (such as filers) • Relationship between local file reference and data stored at object tier is maintained by agent software • Data is moved from local to object tier based on policy (scheduled or ad hoc) • Retrieval of data when client requests access is automatic and transparent

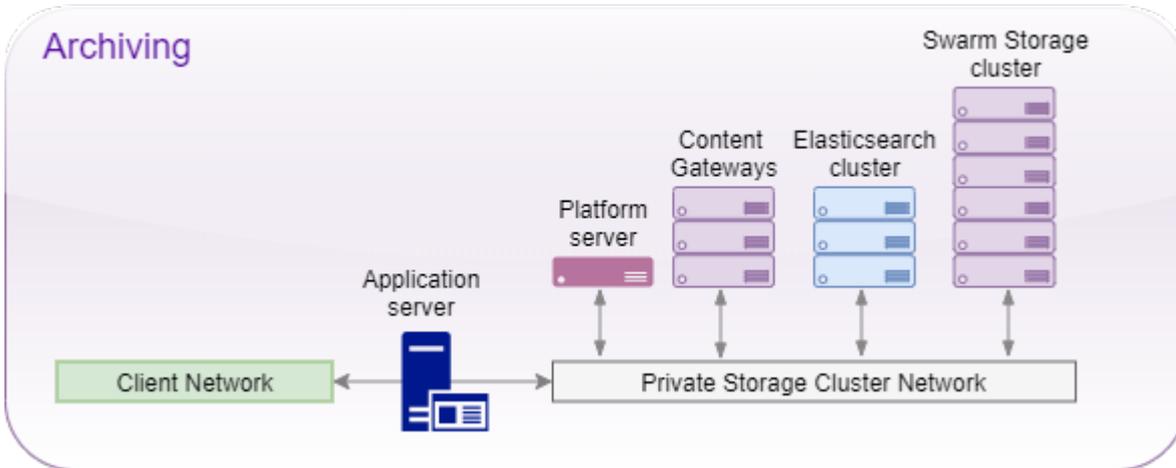
These are common architectures for object storage:

- [Archiving](#)
- [Data tiering](#)
- [Remote replication and disaster recovery](#)
- [Managed service \(“Storage as a Service”\)](#)
- [Hybrid Cloud \(local storage with Cloud\)](#)

Archiving

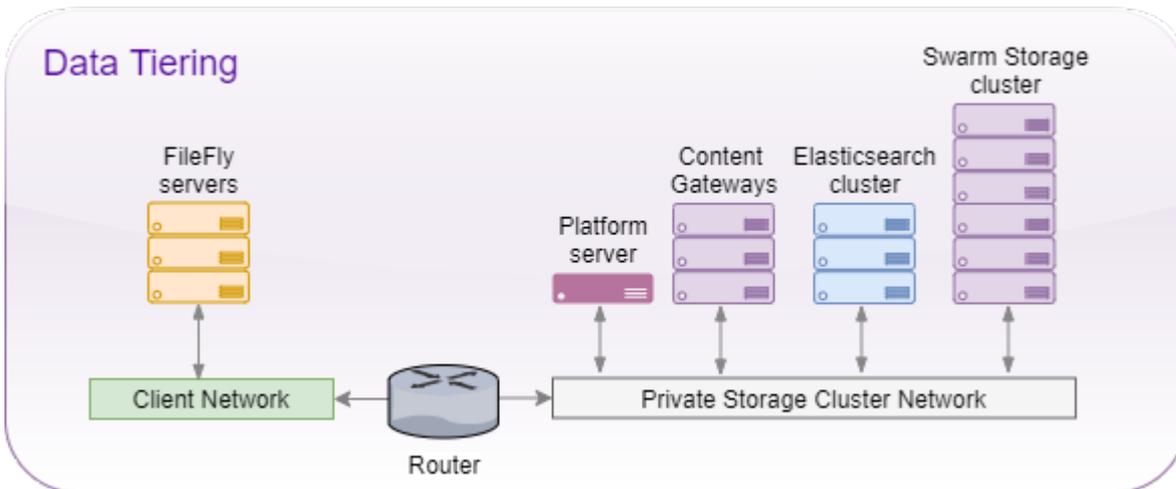
- Medium- to long-term storage

- “Write once, read rarely”
- Library of unstructured data (documents, graphics, pictures, videos)
- Query and list, based on metadata tags
- Conduct “Data Lake” analysis (by pooling a vast amount of raw data in a native format)

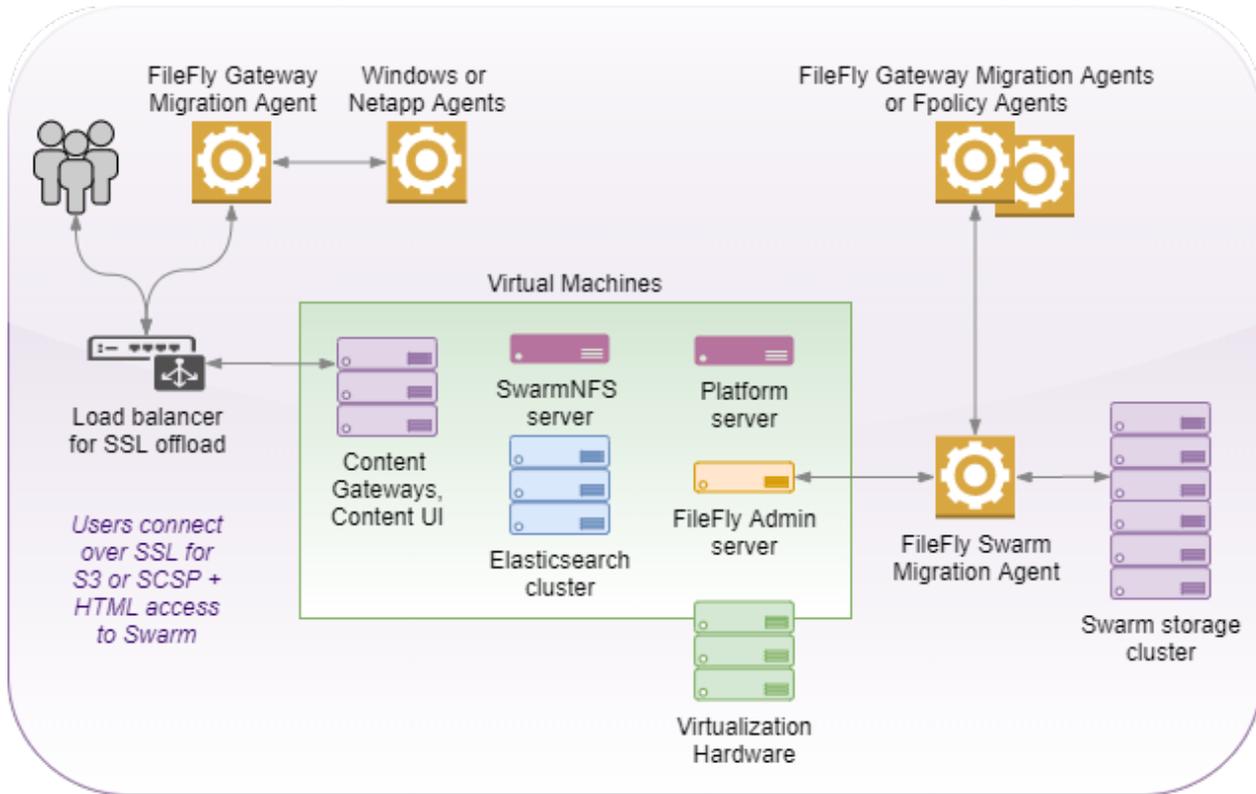


Data tiering

- Relocation of data from traditional filers to object storage
- Scheduled tiering based on policy
- Automated recall when access request is made
- Transparent access to the end user
- “Cheap and deep” object store tier to reduce filer expansion costs

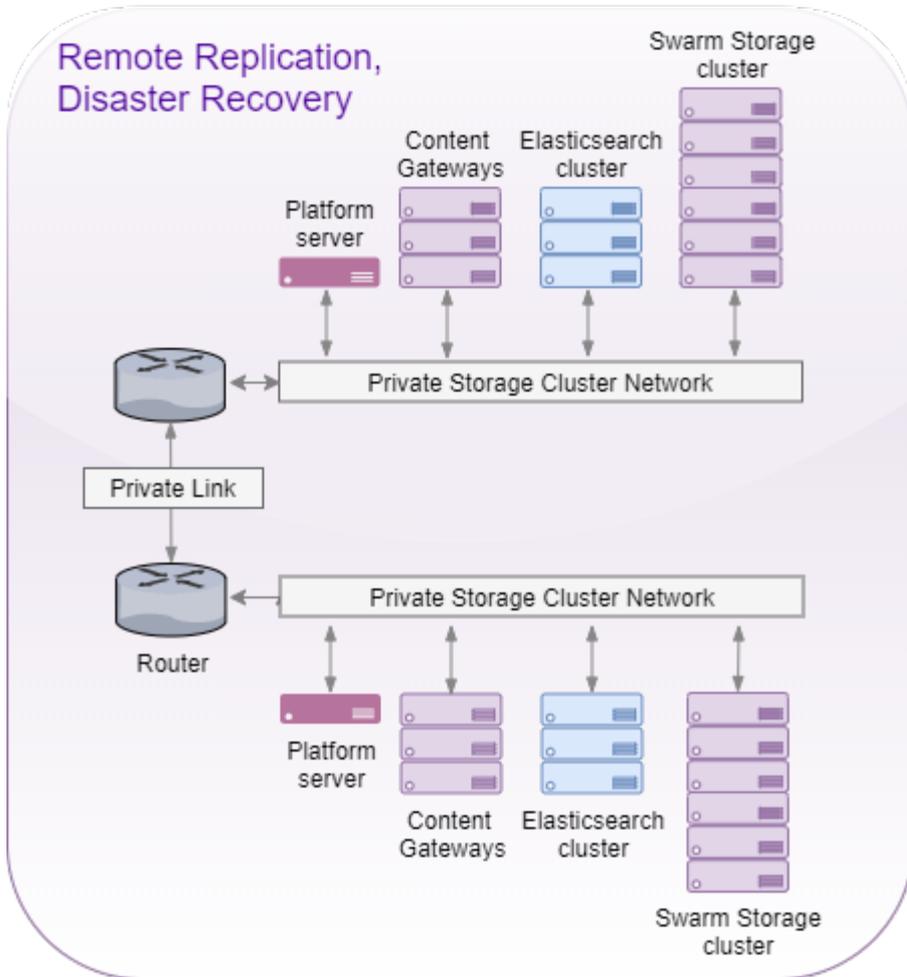


FileFly and Virtualization

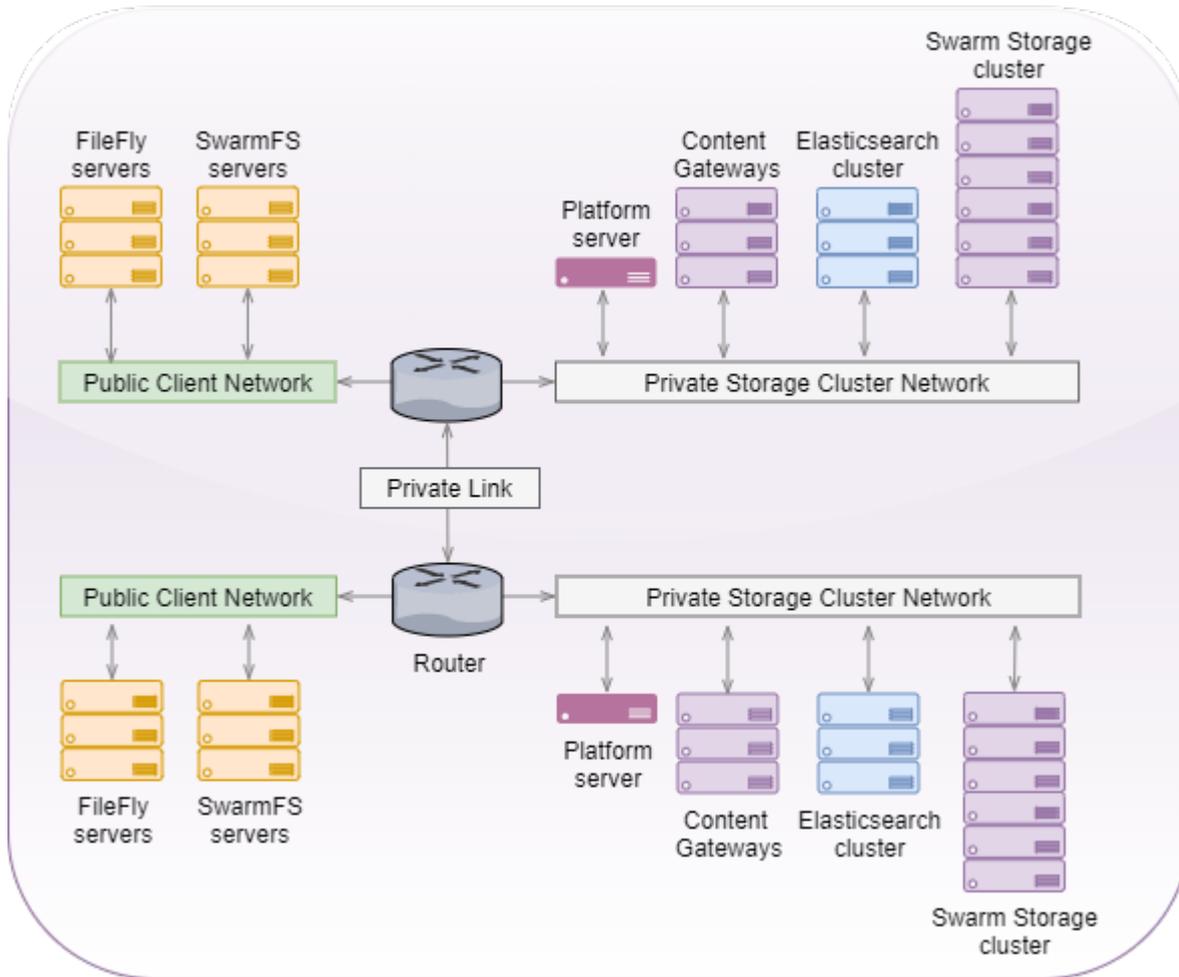


Remote replication and disaster recovery

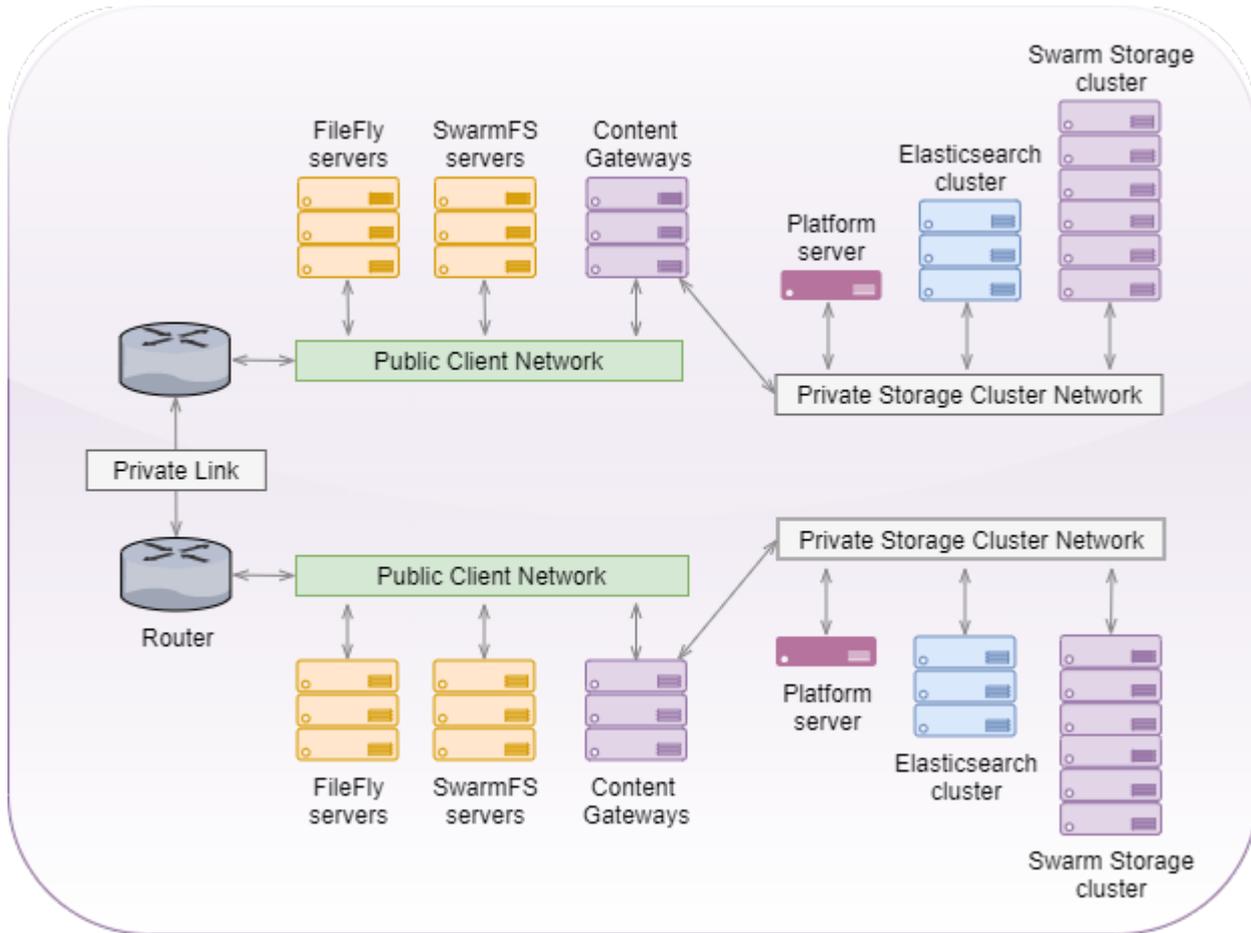
- Automated replication from a local object store to a remote object store
- Data is usually populated in a local store, then replicated to remote/DR
- "Hot" sites can also act as replication targets/DR for each other
- Can be whole site replication or policy based (per domain)
- Varying complexity in replication topologies supported (site-to-site, M to N, single or bi-directional)



Dual Site with Single Interface

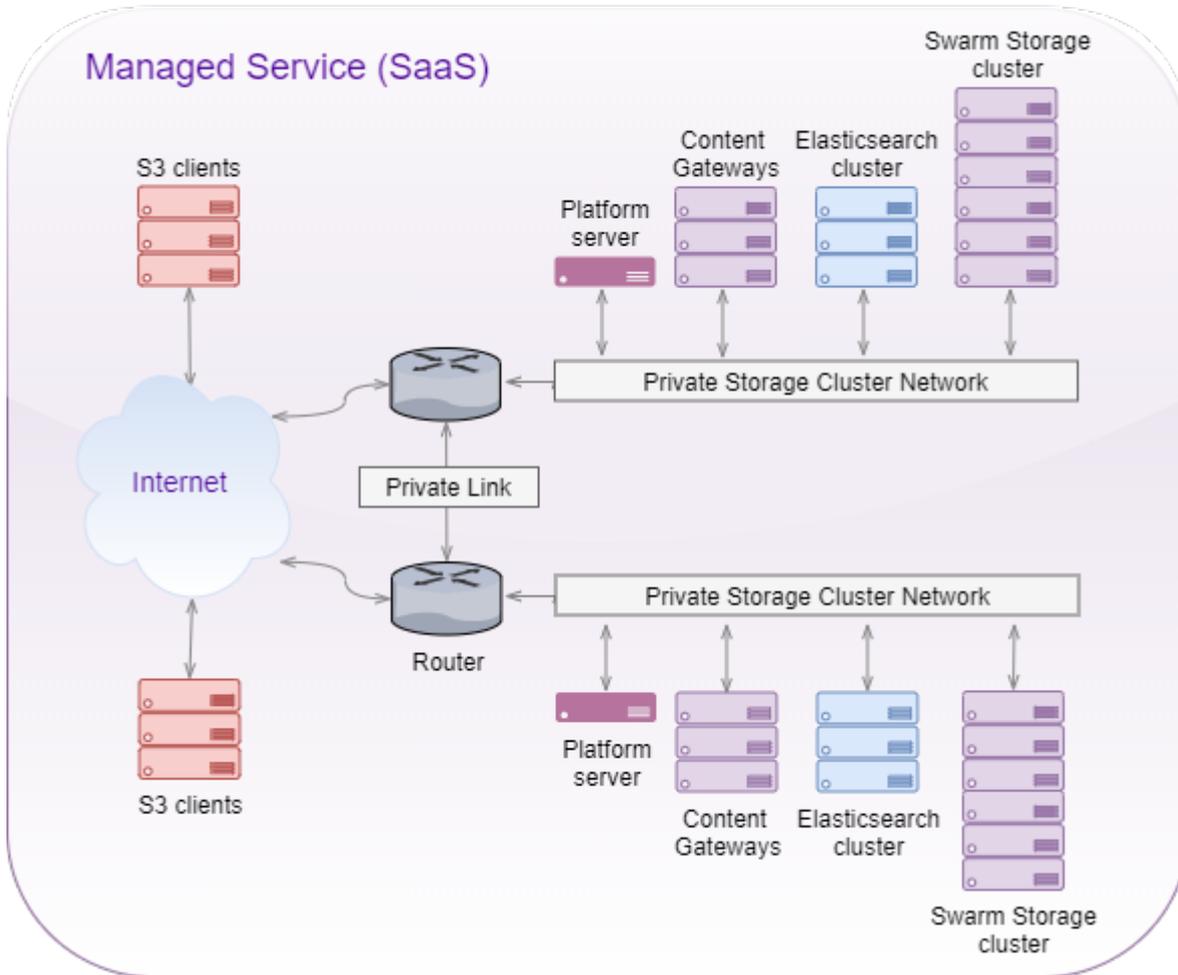


Dual Site with Dual Interface



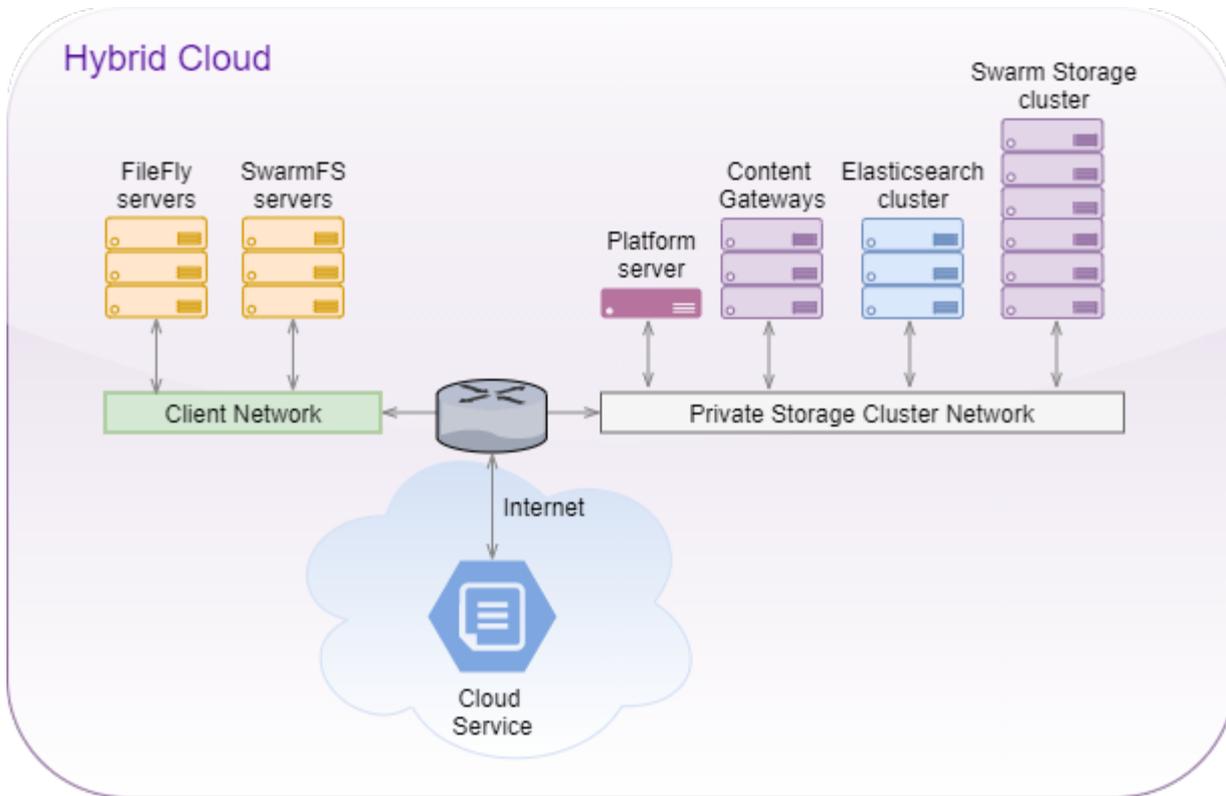
Managed service (“Storage as a Service”)

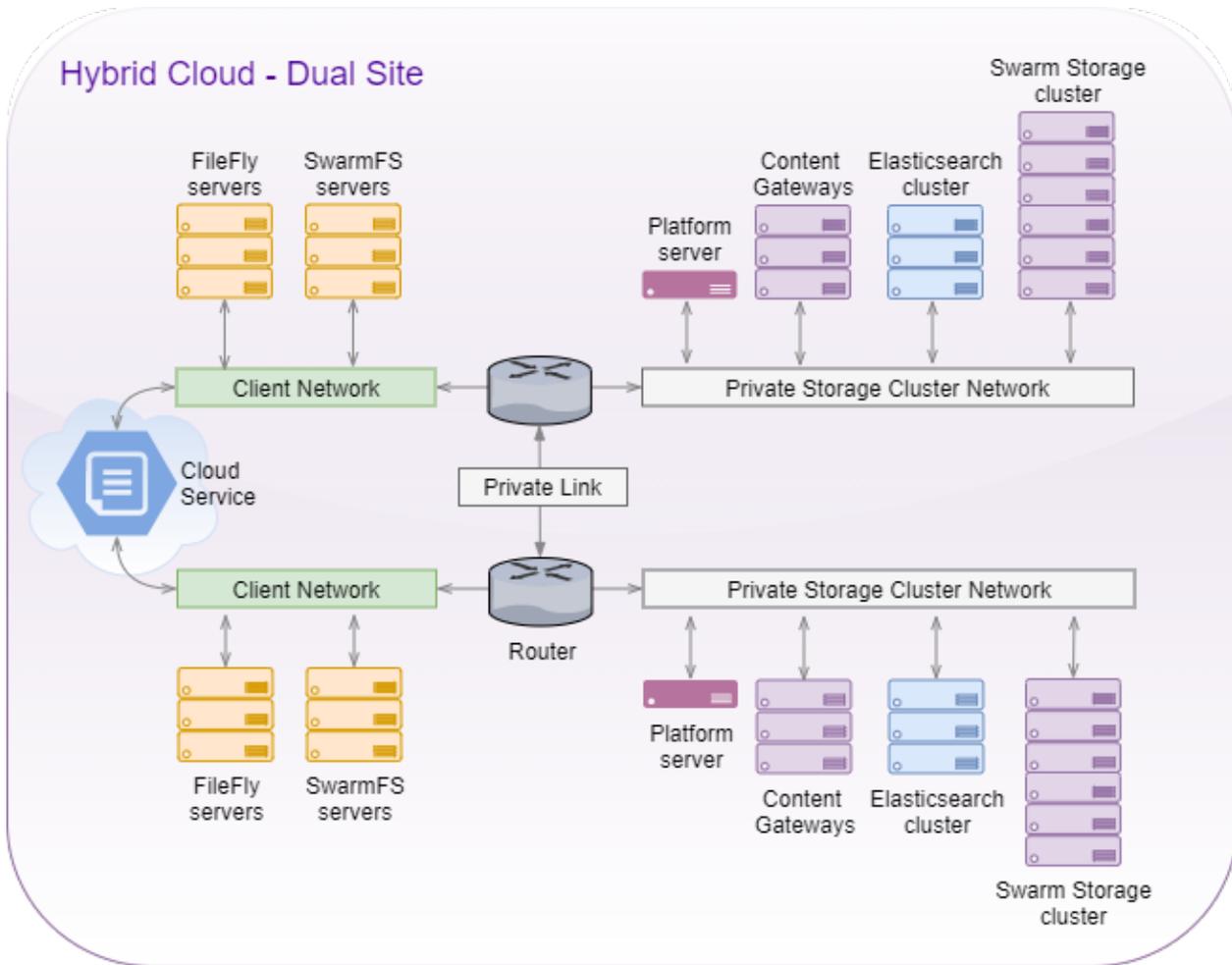
- Storage protocol endpoints made available to service subscribers
- Support for multiple RESTful access protocols
- SSL/TLS
- Provides authentication and authorization
- Allows for metering and billing
- Supports quota control
- Multi-tenancy (individuals, business organizations, business units)



Hybrid Cloud (local storage with Cloud)

- Local object store integrated with a cloud service endpoint (such as Azure)
- “Copy to Cloud” for backup and/or publication of data
- “Retrieve from Cloud” for data recovery
- Lower CapEx when meeting backup/replication/DR requirements





Content UI Installation

- [System Requirements](#)
- [Installing the Content UI](#)
- [Required Access Policies](#)
 - [Essential Permission](#)
 - [Listing Permissions](#)
- [Customizing the Content UI](#)
 - [Tip](#)
- [Upgrading the Content UI](#)

See also [Content UI Release Notes](#) and [Swarm Content UI \(usage guide\)](#).

System Requirements

Hardware	<p>The Content UI does not require additional hardware. Install the Content UI on every Gateway server that services users that access content.</p> <p><i>Exception:</i> There is need to install the Content UI on a Gateway subset if the subset in a load balancing group are reserved for external access (where Content UI access is not required).</p>
Swarm	<p>Swarm Storage must be ready:</p> <ul style="list-style-type: none"> • Be configured to support Gateway • Have a Search feed configured and running
Gateway	<p>One or more Gateway servers must be ready:</p> <ul style="list-style-type: none"> • Be correctly configured and started • Have the SCSP protocol configured for external access • Have idsys.json and policy.json files set for authentication and authorization (see Content Gateway Authentication and Gateway Installation)
Browser	<p>The Content UI requires a JavaScript-enabled internet browser.</p> <p><i>Note:</i> Development and testing are conducted using the most recent versions of Firefox and Chrome.</p>

Installing the Content UI

1. Install and [configure Gateway](#). Verify all storage domains have DNS entries to properly access them from the Content UI.



Important

Any Gateway server on which you are installing the Content UI must be configured with SCSP enabled.

2. Navigate to the directory where Gateway was unzipped.
3. Install the Content UI.

```
yum install caringo-gateway-webui-{version}.rpm
```

i CloudScaler Portal

The Content UI replaces the legacy CloudScaler Admin Portal. If the old Admin Portal was previously installed on the Gateway server, you must remove it because it is incompatible:

```
rm -rf /etc/caringo/cloudscaler/portal
rm /etc/caringo/cloudgateway/web.d/10portal.web.xml
```

4. Restart the Gateway.

```
systemctl restart cloudgateway
```

5. Navigate to the login page for the Content UI, using the base URL of any storage domain in the cluster on the configured SCSP port:

```
http://{storage-domain}/_admin/portal/
```

Required Access Policies

What is visible in the Content UI is controlled and protected by access policy documents. A policy that grants access to a particular domain blocks the members from seeing anything (domains, tenants, clusters) outside of the domain for which they are authorized. Policies must be set to grant use of the Content UI as part of the implementation.

Essential Permission

These are the essential [permissions](#) to allow users with *no other domain-level permissions* to navigate to and view a bucket and objects:

- **GetDomain** is the essential, required permission for all domain users to see the Content UI.
- **GetBucket** is required to display some charts.
- **ListEtc** (List Policies in UI) is needed to see collections listed.
- **GetPolicy** is needed to open a collection.
- **GetQuota** is required to display some charts.

Listing Permissions

- **ListDomain** is required to list buckets in the domain. It also allows SCSP listings and collections to list throughout the domain, regardless of ListBucket permissions.
- **ListDomains** is required to list domains in the tenant.

Note:
This action must be added via JSON text editor prior to Content UI 7.6.0.
- **ListBucket** is required for the listing of objects within a bucket.

See [Setting Permissions](#) (Access Policy editor), [Gateway Access Control Policies](#), and the best practices in [Policy Document](#).

Below is an example tenant policy:

```
{
  "Version": "2008-10-17",
  "Id": "Tenant policy granting all authenticated users read-only access throughout tenant.",
  "Statement": [
    {
      "Sid": "1: Actions required by Portal. No GetObject, no CreateDomain.",
      "Effect": "Allow",
      "Principal": {
        "user": [ "*" ]
      },
      "Action": [
        "GetPolicy",
        "GetTenant",
        "GetDomain",
        "GetBucket",
        "ListEtc",
        "ListTenant",
        "ListDomains",
        "ListDomain",
        "ListBucket",
        "GetQuota",
      ],
      "Resource": "*"
    }
  ]
}
```

Customizing the Content UI

Styling – The Content UI incorporates an empty CSS file used to override the Content UI's styling, both for rebranding purposes and to protect changes across upgrades. The customization stylesheet is `css/custom.css`.



Tip

The `custom.css` includes instructions for how to replace logos on the login page, headers, and **About** page.

Help links – Customize the links for Documentation and Support on the Resource Menu:

1. In a text editor, open `/opt/caringo/gateway-webui/customLinks.json`
2. Locate and add URLs for one or both of these properties:
 - `"userDocumentationLink": ""`
 - `"userSupportLink": ""`
3. Reload the Content UI and verify the "Documentation" and "Online Support" links point to those specified in the JSON file.

Upgrading the Content UI

See [Upgrading Gateway](#) and the [Content Gateway Release Notes](#) for the installed version because the Content UI is upgraded along with Gateway.

- [Troubleshooting UI Upgrades](#)

Troubleshooting UI Upgrades

Users may not see the newer version after upgrading the UIs. Two options to verify are:

- Checking the **About** page (from the global system menu)
- Comparing the view in a browser *Incognito* mode

This is not resolved by logging out, opening in a new tab, or performing a page *Reload*. Instead, direct the users to perform a "hard refresh" to bypass the cache and establish the new version.

To perform a hard refresh

Open each UI application and perform the following:

- **Chrome or Firefox for Windows:** Press `Ctrl+F5` (If that does not work, attempt `Shift+F5` or `Ctrl+Shift+R`).
- **Chrome or Firefox for macOS:** Press `Shift+Command+R`.

Update the browser (**Help > About**) and repeat the hard refresh or clear the browser's data cache through the browser's **Settings** if that does not work.

Swarm Storage UI Installation

This section describes how to install the Swarm Storage UI. To use the legacy Admin Console (<http://{cluster}:90>), no additional installation is required.

See also [Storage UI Release Notes](#) and [Swarm Storage UI](#) (usage guide).

- [System Requirements](#)
- [Installing the Storage UI](#)
 - [Note](#)
 - [Installing on Content Gateway](#)
 - [Installing on CSN 8.3](#)
- [Upgrading the Storage UI](#)

System Requirements

<p>Network</p>	<p>The system where the Storage UI is installed must have direct access to the Swarm Storage VLAN.</p> <p>Adjust firewall rules to allow the direct access required:</p> <ul style="list-style-type: none"> • Swarm Storage nodes (port 91 by default) • Elasticsearch nodes (port 9200 by default).
<p>Swarm</p>	<p>Swarm Storage must be running, with additional configuration to enable the trend charts to display data:</p> <ul style="list-style-type: none"> • A Metrics instance, configured and available. • Elasticsearch must be configured to work with web applications with the following settings: <pre>http.cors.enabled: true http.cors.allow-origin: "*" </pre>
<p>Browser</p>	<p>The Storage UI requires a JavaScript-enabled internet browser.</p> <p><i>Note:</i> Development and testing are conducted using the most recent versions of Firefox and Chrome.</p>

Installing the Storage UI

The Swarm UI supports different installation methods, depending on where it is installed. If you have a Content Gateway implemented, the best practice is to install it there.

Note

The UI defaults to using port 91. If the bind port is changed to anything other than 91, users need to specify it when logging in: `{host} : {custom-port}`

Installing on Content Gateway

The Storage UI installs on a server running the Content Gateway, and it is the recommended implementation. To add the UI to a server running Gateway:

1. Locate the Storage UI RPM in the Swarm bundle.
2. Copy the UI rpm to the directory where the Gateway was installed, and run the following:

```
yum install caringo-storage-webui-{version}.rpm
```

3. To access the UI, browse to:

```
http://{hostname}/_admin/storage
```

Installing on CSN 8.3

See [Scenarios for Swarm UI](#) for how to add Storage UI under different CSN usage scenarios.

To use the Swarm Storage UI with Swarm Metrics, you need all of the enabling components, such as a simplified Gateway (Service Proxy) to provide the needed access and an Elasticsearch service and curator. Your CSN download includes all of the RPMs and scripts needed to perform the installation and configuration of these components.

Caution

- Do not use the CSN-embedded ES server as a Swarm Search Feed target.
- Do not change the preconfigured defaults for Metrics, which are set in `/etc/caringo-elasticsearch-metrics/metrics.cfg`
 - `metrics.target` and `metrics.port` are preconfigured for the Service Proxy.
 - Metrics are checked every 15 minutes and are retained for 120 days.
- If you have any scaling concerns, consult Support about whether you need a dedicated Elasticsearch cluster.

1. Install the CSN bundle as usual (the new RPMs will install for you).
2. Add a Swarm license and start up the Storage nodes.
3. Run the integration script to configure the Service Gateway, adding arguments as appropriate:

```
/usr/bin/configure_storage_webui_with_serviceproxy.py
```

Optional arguments	Default	Purpose
--------------------	---------	---------

-h, --help		Display help summary and then exit.
-v, --verbose	INFO	Add to generate more extensive logging output.
-d DIR, --directory DIR	/var/log /caringo	Specify a different working directory for logs and output.
-s, --start_services	yes	Disable automatic enabling and start up of services after configuration.
-n NUM_HOSTS, --num_hosts NUM_HOSTS	5	Specify a different number of hosts to add to the Gateway configuration.
-indexer_host INDEXER_HOST [INDEXER_HOST ...]		Specify remote Elasticsearch hostnames or IPs, disabling (and preventing creation of) a local ES instance. Important: If you have an existing ES cluster, be sure to specify it here to avoid creating an extra local instance.

4. Each time the script prompts you for a configuration setting, press Enter to accept the suggested value, or enter your own.
5. On completion, the script will attempt to start all of the services with their new configurations, logging the output to the `integration.log` file (which defaults to `/var/log/caringo`).
6. When the script provides you with your URL and credentials, log into the Storage UI with a JavaScript-enabled browser (development and testing were conducted using the most recent versions of Firefox and Chrome).

Note
Metrics data will not appear until new Storage data is added and metrics collection cycles have occurred, which is a minimum of 30 minutes.

7. The script creates a "caringoadmin" user (password "caringo"), which is a user that is local to the CSN. These user credentials are required to login to the Storage UI through the Service Gateway.

Note
This `caringoadmin` user is only used by the Service Gateway, not by the Swarm cluster.

- a. To change the `caringoadmin` password on the local CSN, run the password command and follow the prompts:

```
passwd caringoadmin
```

- b. The user `caringoadmin` user is defined as the administrator for the Service Gateway in this file on the CSN: `/etc/caringo/cloudgateway/policy.json`. The user can be changed according to your requirements, just as in the Content Gateway. See [Gateway Access Control Policies](#).

8. With CSN 8.3, all of these interfaces remain available and may be used concurrently:

Legacy CSN Console	<code>http://{CSN.host}:8090</code>
Legacy Admin Console	<code>http://{CSN.host}:8090/services/storage</code>
Swarm Storage UI	<code>http://{CSN.host}/_admin/storage</code>

Upgrading the Storage UI

Follow this sequence according to the guidance in the [release notes](#) to upgrade the Storage UI:

1. As needed for this release, complete upgrades of Swarm components in this order:
 - a. [CSN Platform Server](#)
 - b. [Swarm Storage](#)
 - c. [Elasticsearch](#) (for S3 and [Swarm Historical Metrics](#)) – Create a new feed, allow it to finish, then [make it Primary](#).
 - d. [Content Gateway](#)
2. Copy the UI rpm to the directory where the Gateway was installed, and run the following:

```
yum install caringo-storage-webui-{version}.rpm  
# If upgrading from 1.1 or earlier, use yum upgrade
```

See [Troubleshooting UI Upgrades](#).

Deployment Planning

- [Environment Planning](#)
- [Swarm Planning](#)
- [VMware Planning](#)
- [Registration](#)

Environment Planning

Research and itemize these environment components before deploying any Swarm Storage solution:

Component	Value	Notes
Enclosures (number)	single dual	Example: Dell PowerEdge M1000e
Storage servers /blades (number)	3 or more	Example: Dell PowerEdge R440
Application servers needing direct cluster access	<i>name, purpose</i>	List all. Example: Enterprise Vault, information governance
Content Gateway (S3) used?	yes no	Both your Content Gateway and your production Elasticsearch cluster need to be on separate machines from your management node (Platform Server or CSN). The management node installs with Service Proxy and a single-node ES, which are dedicated to the Swarm UI.
DHCP server at deployment site?	yes no	
Network Time Protocol (NTP) time server(s)	<i>name /IP address</i>	List all. Use of a local NTP server is best practice
Domain Name Service (DNS) server (s)	<i>name /IP address</i>	List all.
LDAP / Active Directory server(s)	<i>name /IP address</i>	List all.
Timezone of deployment	<u>abbrev, offset</u>	Example: CDT, UTC -5
Default gateway for deployment network	<i>name /IP address</i>	
Subnet mask for deployment network		Example: 255.255.255.0

Swarm Planning

Plan for how the Swarm Storage cluster is configured:

Component	Example	Notes
Name of storage cluster (default domain)	defaultdomain.example.com	Must be a DNS fully qualified name format
Default object replicas	reps = 2	
Default erasure-coding scheme	5:2 for objects > 1 MB	
Cluster replicated?	yes no	
Reserved IPs for Swarm management services	IP1, IP2	Each enclosure needs an assigned IP for this use
Storage be accessed by SSL/TLS?	yes no	
If yes, do you have an SSL/TLS server certificate?	yes no	
Is a load balancer or SSL offload system already in place?	yes no	
Content Gateway use which identity system?	default (PAM) LDAP Active Directory None (anonymous)	
Operating system to install?	CentOS or RHEL	If using RHEL (Red Hat Enterprise Linux), your site needs a Red Hat license

VMware Planning

If using virtual machines in your deployment, research and itemize the following:

Component	Value	Notes
IP address(es) reserved for the ESXi host machine(s)	IP(s)	Must be reserved from the network where the solution is deployed.
System name (FQDN) for vCenter Server Appliance management endpoint		Must be reserved and is required for SSL certificate creation by VMware vCenter.
vCenter Server Appliance management endpoint	hostname / DNS	
Create a vCenter SSO domain or use an existing?	new existing	
vCenter SSO domain		
vCenter SSO site name		

iDRAC gateway	IP	For the integrated Dell Remote Access Controller (iDRAC) interface, enter IP addresses to be assigned for iDRAC access
iDRAC subnet	IP	

Registration

Complete these registrations before proceeding:

1. Register with Support: [Support site](#)
2. Register with VMware, if needed: [vmware.com](https://www.vmware.com)

- [Application and Configuration Guidance](#)
- [Hardware Selection](#)
- [Capacity Planning](#)
- [Deployment Best Practices](#)
- [Cluster Protection Planning](#)
- [Time Synchronization for CentOS7](#)

Application and Configuration Guidance

Follow this guidance when developing Swarm applications or configuring Swarm clusters.

- **Changing Drive Controllers.** Administrators must not move Swarm storage drives between drive array controller types after the drive has been formatted by Swarm. Each controller reports available drive space to Swarm matching the controller. Many controllers claim the last section of the drive, reducing the total available drive space. The new controller may claim additional drive space not reported to Swarm if switching drives with another controller. Swarm may attempt to write data to non-existing drive space, generating I/O errors.
- **Indexer Query Arguments.** The Indexer searching syntax allows for repeated constraints on a field name in the HTTP query string. Verify the HTTP client library is passing all instances of the repeated name and not consolidating the repeats in to one name/value pair if having problems.
- **SNMP behavior with snmpwalk and snmpgetnext.** To be consistent with standard SNMP behavior, the following changes are made to Swarm's SNMP agent:
 - All scalar object IDs (OIDs) end with .0
 - All table OIDs x for row r are returned as x.r from a snmpwalk or snmpgetnext. Custom applications using snmpwalk or snmpgetnext may need to be changed.
- **Known issues with Windows 200x Server time synchronization.** DataCore strongly recommends configuring a cluster to use Network Time Protocol (NTP) as documented about the timeSource. Windows 200x servers as the NTP time source *cannot* be used. Windows servers are not reliable enough to provide highly accurate time synchronization as discussed in [Microsoft KB article 939322](#). consider the following possibilities as an alternative to using time synchronization available in Windows servers:
 - Use NTP servers available on the internet, such as the servers discussed on the [NTP Pool Project page](#). A port in the firewall may need to be opened to enable the cluster to use external NTP servers.
 - Use an open source NTP package such as [the Windows based NTP Time Server Monitor](#).
 - Use a commercial Windows NTP package or deploy a dedicated NTP hardware solution in the network.
- **Virtual Deployments** Administrators wanting to run Swarm in a virtual environment such as VMware must contact a Support representative for restrictions and guidelines prior to deploying Swarm in a VM.
- **Duplicate domain and bucket creation in mirrored clusters.** In a certain cluster configuration referred to as active-active, do not attempt to create the same domain or same bucket in the same domain in each cluster. Instead, create the domain or bucket on one cluster and wait for it to be replicated to the other cluster. Failure to perform this results in the domain or bucket with the latest creation date taking precedence and objects contained in the other domain or bucket being inaccessible.
- **Use curl 7.20.1 or later.** curl 7.20.1 or later must be used if using curl with Swarm, and using the authorization feature. There are known issues with earlier curl versions.
- **Consumer-Grade Drives.** Some non-enterprise-class drives have lengthy error recovery logic. When an error occurs on these types of drives, it may take minutes for a read or write operation to complete. In these cases, the client can see very long response times, or it may even see a socket timeout if the delay is too long. Many enterprise or server grade drives are designed to return errors within a limited period, allowing recovery or rebuild operations to begin immediately and to eliminate the lengthy delays on I/O operations.

Important

Swarm does not support consumer-grade drives in high-demand environments.

- **Avoiding Client Timeouts with Large Objects.** Client operations with large objects (1 GB or greater) can take several seconds or more, depending on object size. Clients supporting large object operations are recommended to set socket timeouts accordingly to avoid client timeout errors.

- **Time Clock Synchronization for Client Servers.** When formatting storage policies in lifepoint headers, it is very important the local clock on the machine creating the lifepoint be reasonably accurate so the end dates of the lifepoints reflect the true UTC time. The Swarm cluster itself can synchronize itself to an accurate time source. If the client mistakenly specifies the wrong end date in an object's storage policy, perhaps because the local clock is set incorrectly, there can be unintended consequences, including premature deletion, when Swarm enforces the policy.
- **Replica Terminology.** The term *replica* has special meaning in the context of Swarm. All instances of an object stored in a Swarm cluster are identical – there is no *original*. Therefore, saying there are two replicas of an object means there are exactly two identical instances (*not* an original plus two copies).
- **Not Found Errors in a Busy Cluster.** A READ, INFO, UPDATE, or DELETE request to a heavily loaded cluster may rarely result in a 404 Not Found response, even if the requested object is present in the cluster. Retry the request until it succeeds if the client has a priori knowledge a certain object is stored. A single retry is usually required.
- **Network Interface Required.** Every Swarm node requires a working network connection. If a network cable is unplugged or if the network is not operational at any time during or after startup, neither SNMP nor SCSP are available. The indication of this condition is in the attached console (if there is one), where errors such as "Network Unreachable" display. Once the connection is restored, Swarm recovers and continues running. Implement SNMP or ping monitoring to verify proper network connectivity among Swarm nodes.
- **HTTP Client Library Limitations.** Some HTTP client libraries, including Microsoft .NET HttpWebRequest and httplib in Python, do not handle the Expect: 100-continue header properly. A client is recommended to include this header when writing content greater than 64 KB and wait after sending the initial headers for a response before sending additional content to Swarm. Possible responses are a redirect (301 or 307), an error response, or the 100 Continue response; continue sending data now. Per the HTTP specification, it is not permissible to continue sending data before receiving a 100-continue response from the server when an Expect: 100-continue header has been included in the request. These issues are resolved in the Swarm Software Development Kit but integrators writing a non-SDK client need to consider these limitations.
- **Available Drive Space.** The available drive space reported by the Swarm Management Console and SNMP is an accurate estimate of the amount of usable space available on a volume or node. The calculation of this value takes in to account a number of internal considerations not immediately visible to an administrator. Swarm reserves space on a volume equal to two times the size of the largest object or EC segment stored on a given volume to allow for continuous defragmentation. The first object or EC segment stored on a volume appears to consume more space than expected. The UUID of the object or EC segment used to reserve defrag space is available in the CARINGO-CASTOR-MIB.
- **Available Index Slots.** The management console may slightly over-estimate the number of available index slots in a node. For capacity planning purposes, use the estimates provided in the memory table.
- **Retire in Small Clusters.** To retire a node or volume, there must be at least two suitable nodes in the cluster having storage space available. Volume-less nodes and nodes retired or retiring do *not* count as suitable nodes. In a multi-server configuration, suitable nodes *can* include other nodes running in the same physical chassis as the retiring node or volume.

Hardware Selection

- [Storage CPU](#)
- [Storage Memory](#)
- [Storage Drives](#)
- [Storage Networking](#)
 - [Best practice](#)
- [Minimum Hardware for Storage](#)
- [Production Hardware for Storage](#)
- [Hardware for Other Components](#)
 - [Virtualization](#)

Storage CPU

- Swarm Storage supports standard x86-64 CPUs (Intel, AMD)
- Single or multiple sockets supported (and multi-core)
- Recommend use of above CPUs that include AES New Instructions (AES-NI) support
 - used by Swarm for improved performance of Encryption at Rest (EAR)
 - most modern server processors include this as of 2010

Storage Memory

RAM per storage node for the following object capacities:

RAM per Node	16 GB	32 GB	64 GB	128 GB
Storage Node RAM index slots	268M	536M	1073M	2146M
Immutable Objects	268M	536M	1073M	2146M
Mutable Objects	134M	268M	536M	1073M
5:2 Erasure Coded Objects	26M	53M	107M	214M

Notes:

- Memory required is a function of object count, object type and data protection scheme chosen
- Larger clusters need additional memory for the Overlay Index or other features which may require additional resources

Storage Drives

- Direct Attached
- Controllers: SAS or SATA JBOD HBAs (SAS preferred)
- “Hot plug” connector / backplane support
- Disks: “Enterprise Grade”
 - designed for 24x7 continuous duty cycles
 - typically 5 year warranty

- Examples: Seagate “Exos”, Western Digital “Gold”

Storage Networking

Best practice

Maintain the same network speed for all devices within the Swarm cluster; mixing speeds requires additional configuration to avoid performance problems.

- Ethernet (with appropriate connector type)
- 1 Gb to 10 Gb (or higher if needed)
- Bonding of multiple ports supported for throughput & redundancy
- including 802.3ad (LAG/LACP) if switch redundancy is required
- Jumbo Frame support
- Typical vendor choices are Intel, Broadcom etc.

Minimum Hardware for Storage

- Appropriate for functional design & testing
- 3 or more nodes (chassis) in a cluster
- Can be deployed as virtual machines (VMware guests)
- Rule of thumb minimum physical memory is 2 GB + (0.5 GB * number of volumes), but more memory improves cluster operation



Production Hardware for Storage

- Multi-socket / Multi-core x86-64 CPUs
- “Enterprise Grade” SAS drives
- RAM depends on object counts and other factors
- Minimum of 4 nodes (chassis) in the cluster (scale up / scale out)
- Typically physical servers, but can be virtual machines (VMware)



Hardware for Other Components

Virtualization

Swarm supports virtualization via VMware ESXi, Linux KVM, and Microsoft Hyper-V (contact DataCore Support for details).

Component	Platform Server	Elasticsearch	Content Gateway	SwarmFS
Purpose	Boot, monitor, manage Storage cluster	Query and list objects in Storage	Protocol and auth/auth gateway to Storage	NFS protocol gateway to Storage
CPU	x86-64 (multi-socket/core, 2 cores)	x86-64 (multi-socket/core)	x86-64 (multi-socket/core)	x86-64 (multi-socket/core, 4+ cores)

Memory	8 GB RAM	64 GB RAM per 1 billion distinct objects	4+ GB RAM	4+ GB RAM (16 GB recommended)
Drive	80+ GB (large clusters: more for logs)	1.5 TB per 1 billion distinct objects	4+ GB plus OS install footprint	40+ GB plus OS install footprint
Network	1 Gb Ethernet	1 Gb Ethernet	1 Gb Ethernet	1 Gb Ethernet (10 Gb heavy traffic)
Servers	1	3 to 4 (for redundancy and performance)	Scale to support client sessions	Scale to support client sessions
Virtualize	Yes (OVA available)	Yes	Yes	Yes
Notes		Assume full index of object metadata (custom metadata)		Scale RAM and CPU with concurrent writes

Capacity Planning

- [Storage Capacity Factors](#)
 - [Expected Object Count and Average Object Size](#)
 - [Choice of Protection Scheme](#)
 - [Need for High Availability](#)
 - [Memory for Overlay Index](#)
- [Elasticsearch \(Search and List\)](#)
- [Gateway \(including S3\)](#)
- [SwarmFS](#)
- [FileFly](#)

Following is a high-level view of factors to consider when researching what hardware capacity is needed for the Swarm implementation.

Storage Capacity Factors

Expected Object Count and Average Object Size

- Object count and object size are the primary drivers for capacity planning
- Object count drives storage cluster memory requirements: more objects requires more memory for the cluster's overlay index
- Average object size multiplied by object count provides the *logical* storage footprint (the amount of content that has been uploaded to the cluster), but it does not account for the space taken by replicas/segments from the protection scheme
- Average object size is the key factor (along with cluster size) for which protection scheme to use (replication vs. erasure coding)

See [Elastic Content Protection](#).

Choice of Protection Scheme

- Which protection scheme chosen drives the memory requirements for the storage cluster
- Erasure coding (EC) requires more memory than Replication (which uses more space)
- Erasure coding impacts CPU performance requirements (because of calculating parity for erasure coding)
- Required volume footprint is derived from combination of (object count) x (average object size) x (protection scheme overhead)
 - *Replication example:* (1 million objects) x (1 megabyte/object) x (2 replicas) = 2 TB
 - *EC example:* (1 million objects) x (1 megabyte/object) x (5:2 EC scheme or 7/5) = 1.4 TB

RAM per Node	16 GB	32 GB	64 GB	128 GB
Storage Node RAM index slots	268M	536M	1073M	2146M
Immutable Objects	268M	536M	1073M	2146M
Mutable Objects	134M	268M	536M	1073M
5:2 Erasure Coded Objects	26M	53M	107M	214M

See [Configuring Content Policies](#).

Need for High Availability

Knowing what failure scenarios can and cannot be tolerated helps with design optimization:

- A requirement for high availability (HA) drives extra capacity needed to cover more catastrophic disk and server failures
- Designs typically account for either multiple volume or multiple server failure scenarios
- Availability requirements vary in complexity, and usually feed back to protection scheme choice

i Best practice

Start expanding the cluster when cluster used capacity reaches 80%.

Memory for Overlay Index

- Other features may be enabled in a cluster which require more resources to properly support them
 - Example: Overlay Index for large clusters (32+ nodes)
- Always consider and account for the resource impact of a given feature/setting before enabling it in a cluster.

i Best practice

Allow an additional 25% of cluster memory to support the Overlay Index.

Elasticsearch (Search and List)

- Provides ability to search for and list objects based on metadata
- Always assume full index of object metadata (custom metadata)
- Memory – 64 GB RAM per 1 billion distinct objects
- Disk – 1.5 TB required for 1 billion distinct objects
- Networking – 1 Gb Ethernet minimum
- Server Count: minimum of 3 to 4, for redundancy and performance
- Scale out as needed by adding more Elasticsearch servers

Gateway (including S3)

- Provides reverse proxy in to storage with added protocol conversion support (S3) and authentication & authorization policy enforcement
- Best treated with a “scale out” approach (think “web farm” behind a load balancer)
- Underlying engine is Java (Jetty)
- Tuned out of the box to account for large session counts based on field feedback
- Memory/CPU/Disk requirements are light for single Gateway server (4 GB RAM/multi-core x86-64/4 GB Disk)
- Networking needs to align with choice used for Storage Cluster (use 10Gb interfaces for the Gateway servers if the Storage Cluster is using 10 Gb interfaces)

SwarmFS

- Provides a protocol gateway for NFS clients (NFS v4.1 to SCSP+)
- Resource requirements are primarily driven by level of concurrent write requests
- Best practice: split up differing NFS client workloads across multiple SwarmFS servers (“scale out”)

- Memory/CPU/Disk requirements are somewhat higher than Gateway (recommended baseline of 16 GB RAM/multi-core x86-64/40 GB Disk)
- As with Gateway, networking choices need to align with the Storage Cluster choice to guarantee throughput

FileFly

- Provides a transparent tiering mechanism to move data from Windows or NetApp file servers into a Swarm storage cluster
- Deployments can range from “single server” configurations to multi-server/high-availability architectures
- Agent software has a small footprint (minimal servers require 4 GB RAM, x86-64 CPU, 2 GB Disk for logs, etc.)
- Treat as a “scale out” solution to support multiple Windows/NetApp file servers (multiple migration agents, multiple fpolicy servers)
- Verify the servers under FileFly source management are “close” to Swarm on the network (avoid routing)
- Align network interface choice for FileFly components with those used in Storage Cluster for best throughput/latency characteristics
- Note: sources under FileFly management tend to become “oversubscribed” (i.e., more data associated with the source server exists in Swarm than can be held locally by the source server)
- Capacity planning for the FileFly source servers becomes important when wanting to perform a large de-migration from Swarm
- Verify this scenario is planned for when assigning storage shares from the source servers to clients

Deployment Best Practices

- [Top-Level Planning](#)
- [Storage Cluster Best Practices](#)
- [Elasticsearch Best Practices](#)
- [Gateway / S3 Best Practices](#)
- [SwarmFS Best Practices](#)
- [FileFly Best Practices](#)

Following are a collection of best practices and reminders for various stages and areas of a Swarm implementation.

Top-Level Planning

- Verify all network requirements are met
- Configure the Switch/VLAN, such as IGMP Snooping and Spanning Tree
- Decide IP assignments, both cluster and client-facing
- Assign Multicast Group
- Create a detailed diagram of the intended implementation (see [Use Cases and Architectures](#))
- For cluster naming and domains, use IANA FQDN format (`cluster.example.com`), and align them with DNS
- Follow conventions for SSL/TLS certificates
- Plan Authentication/Authorization and the user store (LDAP, AD, PAM, Tokens) (see [Content Gateway Authentication](#))
- Decide how data is segmented across tenants, domains, and buckets (see [Migrating from Traditional Storage](#))
- Define policies for client access and data flow; which clients are used to create and access data?
- Choose an approach to integrate clients and applications; are there multiple access protocols to the same data (namespace)?

Storage Cluster Best Practices

- Itemize and account for performance requirements, if any
- Plan for both *maintenance* (drive replacement, live upgrades) and *disruption* (server failure, drive failure) scenarios
 - Verify protection scheme choices align with available resources
- Select both monitoring and notification approaches
- Capture utilization trends to stay ahead of capacity planning (increasing hardware *and* licensing level)
- Create a default domain in the cluster having the *same name as the cluster name* (this is the “catch all” for enforcing tenancy of objects in the cluster)
- Verify all domains in the cluster use IANA FQDN format, as this has ramifications for DNS, Gateway, S3, and SSL+TLS

See [Storage Implementation](#).

Elasticsearch Best Practices

- Plan to “scale out” alongside Swarm Storage
- For best performance and redundancy in production, start out with four ES servers
- Allow no Elasticsearch server to go beyond 64 GB of physical memory (this affects Java max heap and performance)
- To optimize listing and query performance, use SSD drives
- Locate Elasticsearch servers on the same subnet as Storage Cluster (avoid routing to Swarm nodes)

- Read the release notes for Storage Cluster regarding associated Elasticsearch changes which may be necessary when performing upgrades
- Always use the Elasticsearch packages bundled with the Swarm version deployed

See [Elasticsearch Implementation](#).

Gateway / S3 Best Practices

- Gateway serves as a “scale out, lightweight reverse proxy” to object storage
- Place multiple Gateway servers behind the load balancer
- Perform SSL/TLS off-load at the load balancer layer
- Verify Gateway servers have unfettered access to Storage Cluster and Elasticsearch nodes
- For best performance, place Gateway servers on the Storage Cluster network
- Verify Gateway is provided access to LDAP/AD targets that are “network close” (as few hops as possible) and in good working order
- Monitor concurrent session count for Capacity Planning; heavy S3 request activity may require additional Elasticsearch resources

See [Content Gateway Implementation](#).

SwarmFS Best Practices

- Stateless Protocol Translator from NFSv4 to Swarm (SCSP)
- Run the latest Swarm version for best performance
- Scale out vs. export count (memory) and exports that exhibit large concurrent access activity
- Verify SwarmFS deployment planning aligns with authentication and authorization approach for Swarm Storage (Anonymous / Single User / Session Token)
- Verify NFS clients can use NFSv4 (other NFS versions not supported)
- For best behavior, clients should mount SwarmFS exports using a “timeo” setting of 9000

See [SwarmFS Deployment](#).

FileFly Best Practices

- Use named servers (DNS, FQDN) rather than IP addresses, so future server migrations are easier when configuring FileFly
- Enable both header options, *Include metadata HTTP headers* and *Include Content-Disposition*, which allows full metadata capture (such as for creating Collections from FileFly data) when installing the FileFly plugins
- Deploy FileFly using Gateway (aka CloudScaler) rather than Direct to Swarm if possible
 - Gateway allows for authenticated access and data segmentation / policy protection of FileFly data
 - Gateway also supports SSL/TLS encapsulation of data in transit
- With Scrub tasks (which cleans Swarm of data no longer associated with a FileFly source), verify the grace period aligns with the overall backup policy
- After performing any data migration tasks, always run a “DrTool from Source” task
 - Running the tool is necessary to verify up-to-date recovery of stubs, which might be accidentally deleted
- FileFly can be sensitive to network throughput, so keep the associated source and target systems as “close” on the network as possible, and use the highest bandwidth available
- Make note of the location of the FileFly logs, for troubleshooting

Cluster Protection Planning

- [Requirements and Guidelines](#)
- [Choosing EC Encoding and Sizing](#)
- [Optimizing Erasure Coding](#)

With Swarm's density-friendly architecture (introduced in Swarm 10), Swarm's cluster structure and protections have changed:

- **Node = chassis.** Swarm is no longer multi-process, because the new architecture assigns one and only one IP address to each chassis, simplifying management.
- **Fewer nodes.** With one Swarm node per chassis (physical or virtual machine), clusters are now *smaller*, in terms of the number of Swarm processes.
- **No auto-subclusters.** Automatic subclustering by chassis is no longer needed, but you can keep using explicit (named) subclusters for optimizing protection across specific locations or networks.
- **Multiple segments per level.** By default, Swarm allows segments to double up per level if needed, deprecating the old setting `ec.subclusterLossTolerance`.
- **Settings Checker.** To ease migration and upgrades, Swarm has a [Storage Settings Checker](#) to run before installation, to identify settings issues to resolve with Support.
- **Cluster-in-a-box.** Swarm supports a cluster-in-a-box configuration by requiring there to be at least 3 nodes in VMs or containers, each with its own IP address and memory index to keep track of replicas.

Requirements and Guidelines

Observe the following data protection requirements and guidelines when designing the Swarm cluster:

- **Small clusters** – Verify the following settings if running 10 or fewer Swarm nodes (do not use fewer than 3 in production). *Important:* If you need to change any, do so *before* upgrading to Swarm 10.
 - **policy.replicas** – The `min` and `default` values for numbers of replicas to keep in the cluster must not exceed the number of nodes. For example, a 3-node cluster may have only `min=2` or `min=3`.
 - **EC encoding** – For EC encoding, verify you have enough nodes to support the cluster's encoding (`policy.ecEncoding`). For EC $k:p$ encoded writes to succeed with fewer than $(k+p)/p$ nodes, use the lower level, `ec.protectionLevel=volume`.
 - *Best practice:* Keep at least one physical machine in the cluster beyond the minimum number needed. This allows for one machine to be down for maintenance without compromising the constraint.
- **"Cluster in a box"** – Swarm supports a "cluster in a box" configuration as long as that box is running a virtual machine host and Swarm instances are running in 3 or more VMs. Each VM boots separately and has its own IP address. Follow the recommendations for small clusters, substituting VMs for nodes. If you have two physical machines, use the "cluster in a box" configuration, but with 3 or more, move to direct booting of Swarm.
- **Subclusters** – All nodes remain in the single, default subcluster unless you manually group them into named subclusters by setting `node.subcluster` across your nodes. Do this if you want Swarm to distribute content according to groupings of machines with a shared failure mode, such as being in the same building in a widely distributed cluster. (Setting `ec.protectionLevel=subcluster` *without* creating subclusters cause a critical error and lower the protection level to 'node'.)
- **Replication** – For data protection reasons, Swarm does not store multiple replicas of an object on the same node. If using fewer physical machines than are required for the replication scheme, use a virtualization/containerization technology to run multiple Swarm nodes on the same hardware appliance. *Do not specify too many replicas:* setting the number of replicas equal to the number of storage nodes can lead to uneven loading when responding to volume recoveries.
- **Erasure-coding** – Best practice is to use `ec.protectionLevel=node`, which distributes segments across the cluster's physical/virtual machines. Do not use `ec.protectionLevel=subcluster` unless you already have subclusters defined and are sure enough nodes (machines) exist to support the specified EC encoding. The lowest level, `ec.protectionLevel=volume`, allows EC writes to succeed if you have a small cluster with fewer than $(k+p)/p$ nodes. See the next section for details.

Choosing EC Encoding and Sizing

The EC encoding defines the way Swarm divides and stores large objects:

- **k:p** – Defines the *encoding*, where
 - **k** (*data segments*) drives the footprint – An EC object's data footprint in the cluster approximates this value: $size * (k+p) / k$
 - **p** (*parity segments*) is protection – Choose the protection level needed, 2 or higher; p=2 and p=3 are most common.
 - **k+p** (*total segments*) is the count of segments – The original object can be reconstructed if any p segments are lost.
- **Manifests** – Segments are tracked in a *manifest*, which is itself protected with p+1 replicas, distributed across the cluster.
- **Sets of sets** – Very large EC objects (or incrementally written objects) are broken up into multiple EC sets, because any segment that's over the size limit triggers another level of EC. Each set has its own k:p encoding, and the overall request combines them all in sequence.

See [Content Protection with Erasure Coding](#).

How many nodes are needed in the cluster depends on both the encoding scheme and the protection profile you are targeting:

EC Profile	Formula	Example: 5:2	Notes
Manifest minimum	p+1	$2 + 1 = 3$	Base requirement for storing manifests.
Segment minimum	ceil((k+p)/p)	$ceil((5 + 2) / 2) = 4$	Objects can be read (but not written) if one node is lost or offline. Per 5:2, 4 nodes allows 2+2+2+1 segment distribution because Swarm allows 2 segments per node. The <i>ceiling</i> (ceil) means the integer that is greater than or equal to the result.
Recommended protection	ceil((k+p)/p + p)	$ceil((5 + 2) / 2 + 2) = 6$	Objects can be read and written if one node is lost or offline. The <i>ceiling</i> (ceil) means the integer that is greater than or equal to the result.
High protection	k+p	$5 + 2 = 7$	Objects can be read and written even if 2 entire nodes are lost or offline.
High performance	(k+p)*2	$(5 + 2) \times 2 = 14$	Recommended for best performance and load distribution (load-balancing becomes easier as clusters expand).



EC Protection Level

This table assumes the default EC Protection Level is used, which is node-based (`ec.protectionLevel = node`). If using subcluster or volume protection, adjust these formulas to that level. See [Configuring Cluster Policies](#).

Optimizing Erasure Coding

What improves EC performance?

1. **Good-enough encoding.** Do not over-protect. The more nodes involved, the more constraints on EC writes to succeed and the more overhead is created:
 - Keeping $k+p$ small reduces the overhead of EC writes.

- Keeping k small reduces the overhead of EC reads.
2. **Consistent scaling.** Rule of thumb: To scale erasure coding, add 1 additional node for each $\text{ceil}((k+p)/p)+1$ nodes.
 3. **Faster nodes.** As a rule, an EC read/write is limited by the slowest node, and there is significant constant expense to set up connections.
 4. **More nodes.** Having more nodes in the cluster than needed for an encoding allows the cluster to better load-balance.

What helps balancing?

1. **Do not run full.** This is the most important principle, so be ready to proactively scale the cluster. Unbalancing typically happens if a cluster is allowed to fill up before provisioning additional, empty nodes.
2. **More nodes.** Larger clusters have an easier time load balancing, and all nodes do not need to be involved in an EC write. A cluster with $k+p$ nodes fills those nodes at the same rate, but, if a node loses a volume, one node fills faster and stops fully-distributed writes, even though there may be ample space on other nodes.

It takes Swarm a long time to rebalance a cluster that is heavy on EC objects, several times longer than if they are fully replicated, because inadequately distributed EC segments can only be moved by health processors on other nodes, and there are many constraints.

Time Synchronization for CentOS7

Steps to set a server's time source to the NTP server:

1. `timedatectl set-timezone UTC`
2. In `chronyd.conf`, add or modify the following IP address setting.
 - a. When using SCS, use the Swarm-internal IP address.
`server {SCS Swarm-internal IP address} iburst`
 - b. To use a different NTP server:
`server {NTP server IP} iburst`

NOTE:
This must not be a Windows Directory SNTP server.

3. Restart chrony:


```
> systemctl enable chronyd
> systemctl restart chronyd
```

4. Verify the time sources are correct.

```
chronyc sources
```

Verify there is an asterisk (*) next to one of the lines, indicating the time has been synchronized. Example:

```
210 Number of sources = 1
MS Name/IP address Stratum Poll Reach LastRx Last sample
=====
^* telamon.tx.caringo.com 3 8 377 152 +5225ns[+7036ns] +/- 98ms
```

5. Once the time is synchronized, sync the BIOS clock.

```
hwclock --systohc
```

SwarmFS Implementation

SwarmFS is an extensible Swarm-native connector to file-sharing protocols such as NFS and SMB, which allow users to access files over the network as if they are local. As an optional component, it has a separate own distribution packaging and licensing.

- [SwarmFS Deployment](#)
- [SwarmFS Export Configuration](#)
- [SwarmFS Overview](#)
- [SwarmFS Troubleshooting](#)
- [SwarmFS Server Installation](#)
- [Ganesha Operations for SwarmFS](#)
- [SwarmFS Listings](#)
- [SwarmFS Planning](#)

See also the [SwarmFS Release Notes](#).

SwarmFS Deployment

- [Swarm Software Requirements](#)
 - [Best practice for upgrades](#)
- [Implementing SwarmFS](#)
 - [Important](#)
- [Mounting the Exports](#)
 - [Important](#)
 - [NFS v4.1](#)
 - [NFS v4.0](#)

Swarm Software Requirements

Following are the Swarm packages that work with and comprise SwarmFS. Download the latest bundle from the [Downloads section](#) on the [DataCore Support Portal](#).



Best practice for upgrades

Upgrade all Swarm components to the versions included in the Swarm bundle unless older versions of Elasticsearch and Content Gateway need to remain.

This is how packages are named for the components in the table below:

- **Storage:** `caringo-storage-*.rpm`
- **Storage UI:** `caringo-storage-webui-*.rpm`
- **Elasticsearch:** `elasticsearch-*.rpm, elasticsearch-curator-*.rpm`
- **Search:** `caringo-elasticsearch-search-*.rpm`
- **Gateway:** `caringo-gateway-*.rpm`
- **Content UI:** `caringo-gateway-webui-*.rpm`
- **Swarm FS:** `caringo-nfs-*.rpm, caringo-nfs-libs-*.rpm`

Component	Configuration Requirements
Storage	<p>Enable Erasure Coding (EC) – see Configuring Cluster Policies</p> <p>Enable Overlay Index: <code>index.overlayEnabled= true</code> – see Configuring the Overlay Index</p> <p>Enable Replicate on Write (ROW) – see Configuring ROW Replicate On Write</p> <p>Set <code>ec.segmentConsolidationFrequency=100</code> (<i>Caution: Do not allow the cluster to run near capacity.</i>)</p> <p>Set <code>health.parallelWriteTimeout</code> (v10.0+) to a non-zero value, such as 1209600 (2 weeks).</p>
Storage UI	Set the Swarm Search feed to have a 1 second Batch Timeout

Elasticsearch	In the Elasticsearch configuration (<code>config/elasticsearch.yml</code>), make changes needed for SwarmFS: <ul style="list-style-type: none"> • <code>Remove: filter: lowercase</code> (case-insensitive metadata searching in Swarm is incompatible with SwarmFS) • <code>Remove: script.indexed: true</code> (use with ES 2.3.3) • <code>Add: script.inline: true</code> (use with ES 2.3.3 and 5.6.12)
Search	Upgrade to the latest Search RPM when upgrading Storage.
Gateway	SwarmFS can be deployed onto the same Linux server as the Content Gateway.
Content UI	Recommended for viewing and managing objects in the Swarm cluster.
NFS	Do not install SwarmFS server on the same host as Elasticsearch.

Implementing SwarmFS

Important

Complete [SwarmFS Planning](#) before proceeding

For SwarmFS, do the following:

1. Install one or more SwarmFS servers for NFS 4 on designated hardware. See [SwarmFS Server Installation](#).
2. Create the exports needed for the implementation. See [SwarmFS Export Configuration](#).

Tip

The same bucket can be exported more than once, each with values (such as **Read buffer size**) optimized for a type of usage. Then point clients and applications to the share best matching the workload.

1. For functional verification and troubleshooting, create a test domain and bucket and then create an export for that bucket.
2. Conduct basic testing of read, write, and delete using the NFS client mounts for each of the SwarmFS exports.
3. Implement HTTPS in front of the service proxy port to help protect the credentials used to access the Ganesha config file and the file itself: see [Replicating Feeds over Untrusted Networks](#).

Mounting the Exports

Follow these guidelines when mounting the SwarmFS exports:

Important

Do not mount any SwarmFS exports on the Ganesha server in production environments.

SwarmFS allows mounting a share immediately, but there is a grace period (1+ minute) before the content displays. Add `ExportAfterGrace = TRUE;` to the `ganesha.conf` file to prevent content from appearing to be missing on newly mounted shares. (v2.3)

<p>Linux</p>	<p>Mount the exports as normal, with these explicit options:</p> <ul style="list-style-type: none"> • Timeout – Increase the timeout, timeo, to 9000. • Version – To verify mounting using the correct protocol, add the "-t nfs" and "vers=<nfsvers>" options. <i>Best practice:</i> Mount using NFS v4.1. Fall back to 4.0 if the client does not support 4.1. <p>NFS v4.1</p> <pre>mount -t nfs -o timeo=9000,vers=4.1 SwarmNFSServer:/ /mnt/SwarmNFS</pre> <p>NFS v4.0</p> <pre>mount -t nfs -o timeo=9000,vers=4 SwarmNFSServer:/ /mnt/SwarmNFS</pre> <p>Adjust the mount command as needed for the OS version. Specify version this way on Ubuntu 10.04:</p> <pre>mount -t nfs4 -o timeo=9000 SwarmNFSServer:/ /mnt/SwarmNFS</pre>
<p>macOS</p>	<p>Not supported.</p>
<p>Windows</p>	<p>Not supported; Windows has no NFS 4.x client.</p>

SwarmFS Export Configuration

- [Important](#)
- [Important](#)
- [Adding Server Groups](#)
 - [Best practice](#)
 - [Note](#)
 - [Important](#)
- [Adding Exports](#)
 - [Quick Setup](#)
 - [Cloud Security](#)
 - [Tip](#)
 - [Client Access](#)
 - [Permissions](#)
 - [Tip](#)
 - [Using x-owner-meta](#)
 - [Logging](#)
 - [Advanced Settings](#)
 - [Important](#)

The NFS page in the Swarm Storage UI allows creating and managing NFS server groups and exports.



Important

The storage cluster's default domain must be created *before* configuring SwarmFS. This domain has the same name as the `cluster.name` setting's value. The domain can be created with the [Content UI](#) or an HTTP utility like curl (see [Manually Creating and Renaming Domains](#)).

Create separate groups (sets) of SwarmFS that are configured in pools; this enables support for different clients and optimization for different roles. Set some configuration settings locally, to override global configuration settings.

Why have different server groups? These are situations for which it may be ideal to keep groups separate:

- Include DEBUG level logging
- Change the log file location
- Add local resource restrictions
- Change interface or IP address bindings
- Reduce maximum threads or open/concurrent client connections

While every SwarmFS server retrieves the global configuration file stored within Swarm, each server group can optionally override the global settings with a separate configuration file.



Important

Restart NFS services after making any configuration changes. The NFS server does not support dynamic updates to the running configuration.

Adding Server Groups

Server Groups are created with the **+ Add** button at top right.



Best practice

Verify the default domain is specified and the existence of the domain and bucket are defined in the scope before creating a Server Group.

The resulting group is a container for exports sharing a common configuration:

Name

Accounting

Configuration URL

Supply a name, which is a description, when adding a Server Group; the unique identifier is the count (such as / 2, above) at the end of the Configuration URL.

The new group appears at or near the end of the listing, ready to be configured with exports.

Each NFS Server Group has a unique Configuration URL, which can be clicked to view the current export definitions. These are the auto-generated and auto-maintained JSON settings being stored by Swarm for the group.

The configuration is empty until one or more exports is added.

 **Note**

An `sptid` parameter is the encrypted form of a Swarm node IP address, which Gateway uses for request routing. *Remove* the parameter when pasting the URL elsewhere, such as in Ganesha.

 **Important**

Although group configurations may be shared across NFS servers, each server must be configured with *only one group*.

Adding Exports

Listing service: Each export is specific to one and only one Swarm bucket, but clients viewing the mounted directory are able to view, create, and use virtual directories within it via the prefix feature of Swarm named objects (`myvirtualdirectory/myobjectname.jpg`).

Name	Unique name for the export, to distinguish it from the others in Swarm UI listings.
-------------	---

<p>Storage IP (s) or DNS name(s)</p> <p>Name * Accounting Reports</p> <p>Storage IP(s) or DNS name(s) * swarm1.example.com swarm2.example.com</p> <p>Search host(s)</p> <p>Search index</p> <p>Export path * /ACCT_Reports</p> <p>Scope * mydomain.example.com acctreports</p>	<p>The IP address(es) or DNS-resolvable hostname(s) for one or more Swarm Storage nodes.</p> <p>(For backwards compatibility) Optional as of version 3.0. The IP addresses or DNS-resolvable hostnames for one or more Swarm Elasticsearch servers.</p> <p>Note: Both Gateway and SwarmFS use the Primary (default) search feed. If a new feed is made Primary, these servers must be restarted.</p>
---	--

Search index		(For backwards compatibility) Optional as of version 3.0. The unique alias name of the Primary (default) search feed. Locate this value as the Alias field in the primary search feed's definition.
Export path		Case-sensitive. Unique pseudo filesystem path for the NFS export. Cannot be set to a single slash ("/"), which is reserved.
Scope	Domain Bucket	Specifies where the data written via the export is associated: which domain and bucket to use. Important: Verify the existence of the domain and bucket specified here.

Quick Setup

For the remaining setup sections, few changes are usually needed:

- **Cloud Security** – Each export can have different security, to fit the usage.
- **Client Access** – Keep the defaults unless access control needs to be customized.
- **Permissions** – Change `nobody` to `x-owner-meta`.
- **Logging** – Keep the defaults unless directed by Support.
- **Advanced Settings** – Keep the defaults unless directed by Support.

Cloud Security

In a Gateway (Cloud) environment pass-through authentication can be used. Authenticating to Gateway can use the same login and password provided for authentication by the client to SwarmFS. Session tokens (with various expiration times) and single user authentication are available, by login credentials or token.

SwarmFS maintains exactly the same level of object security when accessing or modifying objects through SwarmFS or other protocol such as SCSP, S3 or the SwarmFS (Hadoop). Gateway provides security at domain and bucket level only and objects inherit those security policies, accessibility to all unnamed objects are restricted to that of the user's rights at the containing domain, and restricted to rights set at the containing bucket level for named objects. SwarmFS layers no individual object security (named or unnamed) above that enforceable by Gateway.

Tip

Each SwarmFS export created to use the Content Gateway can have an entirely different security method, as needed by the use case.

Session Token	Token Admin Credentials by Login	User, Password, Expiration
	Token Admin Credentials by Token	Token, Expiration

Single User	Authenticate by Login	User, Password
	Authenticate by Token	Token
Pass-through / None	n/a	

Client Access

This optional section allows access control customization both globally (for this export) and for specific clients.

Access type	<p>Defaults to full read/write access. These other access restrictions are available:</p> <ul style="list-style-type: none"> • All operations (RW) - <i>default</i> • No access (None) • Read-only (RO) • No read/write (MDONLY) - allows listing and metadata updates without access to file contents • No read/write/modify (MDONLY_RO) - allows listing but no metadata updates and no access to file contents
Squash	<p>Defaults to no squashing (allows all user IDs).</p> <ul style="list-style-type: none"> • None - <i>default</i> • Root - squashes the remote superuser (root, uid=0) when using identity authentication (local user is the same as remote user) • All - squashes every remote user, including root.
Squash user id (uid) mapping Squash id (uid) mapping	<p>User ID and Group ID can be set when the NFS server is authenticating users from a different authentication sources and/or it is desired all files have a consistent user/group.</p> <p>Typical situations:</p> <ul style="list-style-type: none"> • All clients are configured to use local password/group files, but SwarmFS through the Content Gateway is configured to use LDAP. • All clients have local password/group files, but some users may not exist on all clients systems or may differ on each client. • All clients have the same users and groups, but they are created in a different order. • All clients authenticate using individual logins/accounts, but it is desired all files have the same consistent owner and group regardless of the user reading or writing the files. • A client mounts the NFS exports as anonymous, but it is desired to have the files presented over the share to all NFS clients to have a consistent UID and GID.
Client(s)	<p>As needed, customize the access for one or more specific clients.</p> <p>Note: These override the settings specified above, if any.</p>

Permissions

Files and directories in a SwarmFS system support standard Unix-style read/write/execute permissions based on the user ID (`uid`) and group ID (`gid`) asserted by the mounting NFS client. The numeric forms of `uid` and `gid` have equivalent human-readable ASCII forms, as given by the Linux `'id'` command:

```
$ id
uid=501(smith) gid=20(staff) groups=20(staff)
```

SwarmFS checks the IDs to verify they have permission to access the objects when users attempt to access files and directories and it uses these IDs as the owner and group owner for any new files and directories they create.

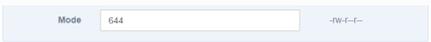
Default **User**, **Group** and ACL **Mode** can be customized for the export mount, directories, and files for each export. These settings only apply for externally created objects and synthetic folders without POSIX permissions attached to the object as standardized metadata. **User** and **Group** values must be entered as ASCII text, not numeric IDs.

Tip

The ACL mode must be entered as an octal, such as 664 or 0664. Use [Chmod Calculator](#) to generate the octal code that corresponds to the read/write/execute permissions to apply.

Using x-owner-meta

The export's interface and access method selected determines whether `x-owner-meta` is used or not. Using defaults of `x-owner-meta` and 0755 or 0644 are valid only when **Storage Interface** is set to "Content Gateway" and the **Cloud Security** Access method is set to "Session Token". For all other methods (such as "Direct to Swarm", "Single User", "Pass-through / None"), the NFS client does not map `x-owner-meta` to a local UNIX/POSIX user.



Logging

Enable additional logging as directed by DataCore Support, but keep this logging disabled for normal production usage. (Swarm UI 2.3)

Performance	Performance logging for SwarmFS, which reduces the noise in the ganesha log file. When enabled, logs PERF warnings to Elasticsearch query result dumps.
Elasticsearch	Performance logging for Elasticsearch, for use while troubleshooting issues such as partial listings. When enabled, sends the Elasticsearch query results to the debug log file.

Advanced Settings

Important

Use these recommended defaults for Advanced Settings unless otherwise advised by DataCore Support.

Transport protocol	TCP	Supported transport protocol (TCP/UDP TCP UDP)
Storage port	80	Required. Network port for traffic to Swarm Storage nodes
Search port	9200	Required. Network port for traffic to Swarm Search nodes
Security	sys	Remote Procedure Call (RPC) security type (sys krb5 krb5i krb5p)

Maximum storage connections	100	Maximum number of open connections to Swarm Storage. (v2.0)
Retries	5	(positive integer) How many times SwarmFS retries unsuccessful requests to Swarm and Swarm Search before giving up.
Retries timeout	90	(seconds) How long SwarmFS waits before timing out Swarm retries.
Request timeout	90	(seconds) How long SwarmFS waits before timing out Swarm requests. For best results, set this timeout to at least twice the value of the Storage setting scsp.keepAliveInterval .
Pool timeout	300	(seconds) How long discovered Swarm storage nodes are remembered.
Write timeout	90	(seconds) How long SwarmFS waits for a write to Swarm to complete before retrying.
Read buffer size	128000000	(bytes) Defaults to 128 MB, for general workloads. The amount of data to be read each time from Swarm. If the read size buffer is greater than the client request size, then the difference is cached by SwarmFS, and the next client read request is served directly from cache, if possible. Set to 0 to disable read-ahead buffering. Improving performance – Set each export's Read Buffer Size to match the workload expected on <i>that</i> share. <ul style="list-style-type: none"> • Lower the read-ahead buffer size if most reads are small and non-sequential. • Increase the read-ahead buffer size if most reads are large and sequential.
Parallel read buffer requests	4	(positive integer) Adjust to tune the performance of large object reads; the default of 4 reflects the optimal number of threads, per performance testing. (v2.3)
Maximum part size	64000000	(bytes) How large each part of erasure-coded (EC) objects may be. Increase (such as to 200 MB, or 200000000) to create smaller EC sets for large objects and so increase throughput for high volumes of large files. (v2.3)
Collector sleep time	1000	(milliseconds) Increase to minimize object consolidation time by directing SwarmFS to collect more data before pushing it to Swarm, at the expense of both RAM and read performance, as SwarmFS slows clients when running out of cache. Increase this value if the implementation is sensitive to how quickly the Swarm health processor consolidates objects, which cannot be guaranteed. (v2.3)
Maximum buffer memory	2000000000	(bytes) Defaults to 2 GB. Maximum limit that can be allocated for the export's export buffer pool. Once exceeded, client requests are temporary blocked until total buffers falls back below this number. (v2.0)
Buffer high watermark	1500000000	(bytes) Once the allocated export buffers reach this watermark, SwarmFS starts to free buffers in an attempt to stay below "Maximum Memory Buffers". During this time, client requests may be delayed. (v2.0)

File access time policy	"relatime"	Policy for when to update a file's access time stamp (atime). (v2.0) <ul style="list-style-type: none"> • "noatime": Disables atime updates. • "relatime": Updates atime if it is earlier than last modified time, so it updates once after each write. • "strictatime": Updates atime on every read and close.
Elasticsearch buffer refresh time	60	(seconds) How rapidly non-SwarmFS object updates are reflected in SwarmFS listings. Lower to reduce the wait for consistency, at the cost of increased load on Elasticsearch. (v2.3)

SwarmFS Overview

- [Simplified Architecture](#)
 - [High Availability](#)
 - [Note](#)
- [Simplified Security](#)

SwarmFS is a lightweight protocol converter that seamlessly integrates Swarm scale-out object storage with [NFS v4](#). It combines Swarm's universal, multi-protocol namespace with the power of enhanced and custom metadata, offering you new ways to manage, view, and analyze your data. SwarmFS provides a traditional file interface to Swarm object storage for content generators (enterprises, researchers, web-based applications, and developers) who use native NFS-based applications to create, access, and manage that content while allowing for the same content to be created, accessed, and managed through modern cloud and object APIs such as S3 and SCSP.

SwarmFS avoids the common problems of traditional gateway and connector file-based storage solutions: protocol and storage silos, bottlenecks, and single points of failure. SwarmFS with Swarm object storage provides, as standard, high availability (HA), data management (from creation to expiration), powerful metadata management, and ad hoc search to your content.

Key features

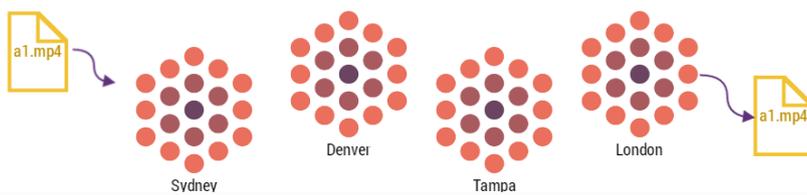
- Universal namespace – eliminates protocol silos by enabling every object to be written/accessed via NFSv4, S3, or SCSP/HTTP, all without restriction
- Lightweight for rapid deployment (VM or physical)
- Built-in active/active HA, no clustering required or limitations on number of active access points
- Direct access to read and update an object's metadata over NFS
- Stateless – reducing data loss of uncommitted resources
- Data is read from and written directly to Swarm. There is no latency and risk of data loss introduced through staging data, as with traditional gateways and connectors
- Leverages Swarm's built-in distributed features for resilience and HA

Benefits

Productivity	Store, access, and manage files	<ul style="list-style-type: none"> • Data portability – multi-protocol in/out through NFSv4, S3, HDFS, or HTTP; read data written through FileFly • Streams data directly to and from Swarm (no local gateway storage staging or spooling) • Brings rich custom object metadata to file though NFS • Mount domains, buckets, or views that are filtered by custom object metadata
Less Risk	Security and scale with no single point of failure	<ul style="list-style-type: none"> • Limitless scale <ul style="list-style-type: none"> • Rapid scale thought physical servers, VMs or appliance • No storage or protocol silos • No read performance performance latency through data staging • Multi SwarmFS instance managed through a single pane of glass • Security settings in Swarm propagate through all protocols • Builtin active/active HA that requires no local disk and no clustering • Auto client resume – if there is a communication issue between client and SwarmFS, the client restarts up where it left off

<p>Lower TCO</p>	<p>Leverage Swarm scale-out storage</p>	<ul style="list-style-type: none"> • High availability and data protection are standard automated features • Continuous protection with seamless movement between replication and erasure coding • Eliminates the need for backups • Leverage Swarm’s automated, policy-based data management • Automatically replicate content to a remote site for distribution • Manage files from creation to expiration • WORM, Legal Hold, and Integrity seals are standard
-------------------------	---	--

SwarmFS provides true distributed file management, with the self-healing modularity of clusters that can have resources come online and go in and out of service with no disruption to user experience:



Simplified Architecture

SwarmFS is a lightweight nfs-ganesha plugin. A traditional gateway or connector has many more moving parts, each adding restrictions, overhead, and complexity:

i

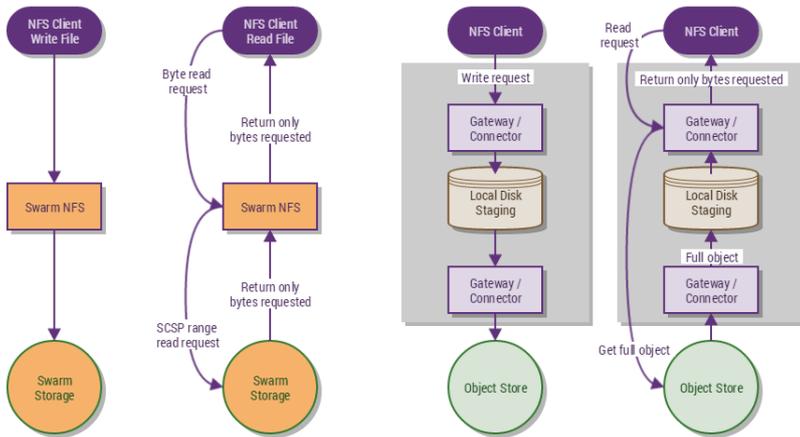
SwarmFS is stateless, so as many SwarmFS instances as needed can be spun up and all NFS instances are active, with no failover or clustering required. High Availability is part of the standard product and no special configuration, clustering or management is required.

Traditional gateways and connectors must stage objects as files locally on the gateway. Disk space on gateway server is a limitation. SwarmFS does not cache or stage files/objects on local disk space; rather, it streams data directly to and from Swarm. You escape the performance overhead of writing complete objects to a local gateway staging disk, and you have no risk of losing data if the Gateway crashes before data is spooled off the gateway to the object store.

i

Note

Given the stateless nature of Swarm and SwarmFS, file locking exists only within a single SwarmFS server.



Simplified Security

POSIX – SwarmFS adds basic POSIX security to the modern object security inherent in Swarm. Uniform object security is achieved through and across NFS, S3, and S3, with *additional* POSIX file security when objects are accessed through SwarmFS.

Access Control – SwarmFS supports basic POSIX UNIX-style ACLs (user, group, other); object access control is managed via [Content Gateway](#). SwarmFS validates the login/password, and it then leaves Gateway to control object access. Native Swarm access means Anonymous NFS access to objects only.

SwarmFS Troubleshooting

- [Required](#)
- [General Troubleshooting](#)
- [SELinux Status](#)
- [Persistent .nfsXXXX Files](#)
- [Changing Logging Levels](#)
- [Failure to Load Export Configuration](#)
- [Client Mounts with Empty Listings](#)
- [Listing Exports and Clients](#)
- [Matching Requests between SwarmFS and Storage Logs](#)
 - [Caution](#)
- [Missing Custom Header](#)
- [Users Lost Permissions](#)
- [Performance Issues](#)



Required

To use `ganesh_mgr` for these troubleshooting steps, first install the RPM package `nfs-ganesh-utils`.

General Troubleshooting

Start with this to begin: Can a client *inside* the ACL mount successfully?

```
mount server:/export/nfs /export/nfs
```

Check these things if not, or if receiving a **permission denied** error:

1. Is `iptables` is allowing access through the firewall, if any?
2. Is SELinux is blocking access? (see next section)

```
/usr/sbin/setroubleshootd
grep httpd /var/log/messages
```

3. Are the `portmap` and `nfs` services running?
4. Can NFS statistics be viewed through `nfsstat`?
5. Can exported file systems be viewed through `exportfs`?

SELinux Status

By default, SELinux does not allow *any* access to remote content. Run this status command to verify SELinux is disabled:

```
sestatus
```

One of these SELinux booleans needs to be enabled if SwarmFS with SELinux enabled is desired to run :

- `nfs_export_all_ro` – allows file systems to be exported read-only
- `nfs_export_all_rw` – allows file systems to be exported read-write
- `use_nfs_home_dirs` – allows home directories to be exported over NFS

Set this with the setsebool utility:

```
setsebool -P nfs_export_all_rw 1
```

Persistent .nfsXXXX Files

Per POSIX standards, Ganesha does not physically delete files that are open at the time they are unlinked. It hides them by a mechanism known as "silly rename": the unlinked files are kept in the same directory but renamed to the form `.nfsXXXX` (with `XXXX` being a random number). These files are cleaned up after the last application using them closes the file handles. These files may linger indefinitely if for some reason this does not occur.

Add a cron job that periodically looks for and deletes such files to verify no "silly" files persist and consume storage space.

Changing Logging Levels

SwarmFS logs to `/var/log/ganesha.log` by default. The logging level for SwarmFS defaults to `NIV_EVENT` to optimize read performance.

Find level – Run the appropriate command to determine the current log level for the SwarmFS plugin or all Ganesha components:

```
ganesha_mgr get_log COMPONENT_FSAL
ganesha_mgr get_log COMPONENT_ALL
```

Change level – Edit the `/etc/sysconfig/ganesha` file to change the logging level permanently. These are supported levels:

- `NIV_EVENT` – SwarmFS default, for best performance.
- `NIV_INFO` – Prints all logs below the level, such as `NIV_FATAL`, `NIV_MAJ`, `NIV_CRIT`, `NIV_WARN`, and `NIV_EVENT`.
- `FULL_DEBUG` – Enable for troubleshooting.

Best practice: Enable debug temporarily without restarting Ganesha using these commands:

- **Start debug** – Run the appropriate command to enable debug logging for the SwarmFS plugin or all Ganesha components:

```
ganesha_mgr set_log COMPONENT_FSAL FULL_DEBUG
ganesha_mgr set_log COMPONENT_ALL FULL_DEBUG
```

Note

`COMPONENT_ALL` is the default for components with no individual log level set.

- **Stop debug** – Run the appropriate command to turn off debug logging for the SwarmFS plugin or all Ganesha components:

```
ganesha_mgr set_log COMPONENT_FSAL NIV_EVENT
ganesha_mgr set_log COMPONENT_ALL NIV_EVENT
```

Failure to Load Export Configuration

SwarmFS may not be loading the configuration if, after starting Ganesha, client root export mounts `[mount {server}:/ {/mntpoint}]` list `/bkt,.`

1. Start Ganesha manually in the foreground.

```
ganesha.nfsd -F
```

2. Wait 20 seconds. Expect output similar to the following if all is working:

```
16:42:24,622231160 P 8186-0x7f5843c32100 libswarmio | ### Registering swarmio logsink f
Remove Export with id 1
Remove Export with id 2
Add Export in /etc/ganesha/318a2790-5a1b-11e8-ala-002590eb7394.tmp
Returns: status = True, Done: 1 exports added
Remove Export with id 1
Remove Export with id 3
Add Export in /etc/ganesha/318a459a-5a1b-11e8-a1
```

3. Look for one set of *Remove Export with id 1*, *Remove Export with x*, and *Add Export* for each of the configured exports. Proceed if these complete sets do not display.
4. Verify SwarmFS can retrieve the central configuration:

```
grep Configuration /etc/ganesha/ganesha.conf
```

5. Navigate to **Settings > NFS** and locate the Configuration URL in the Swarm UI:



6. Use cURL to verify the configuration file can be manually retrieved:

```
curl -L -v {URL}
curl -L -v http://172.30.14.151:91/api/nfs/configurations
```

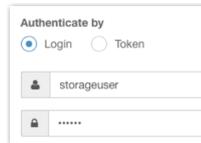
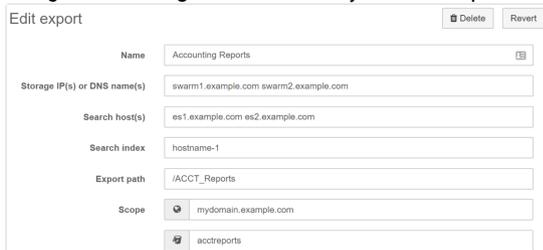
7. Resolve the issue and then restart Ganesha manually in the foreground to verify the configuration file cannot be manually retrieved using cURL.

```
ganesha.nfsd -F
```

Client Mounts with Empty Listings

Follow these steps if client mounts show empty listings (and matching content exists):

1. Navigate to **Settings > NFS** and verify both the export details *and* the authentication in the Swarm UI.



2. Verify the bucket can be accessed using cURL from the SwarmFS server using the configured export details:

```
curl -L -I -v http://{storageip|name}
-u {storageuser}:{storageuserpasswor}

curl -L -I -v http://site.example.com/demobucket?dom:
-u storageuser:xxxx
```

3. Verify Elasticsearch (as defined in the **Search host(s)** export field) can be accessed from the SwarmFS server if the bucket can be accessed:

```
curl http://{search host}:9200/_cluster/health -v
curl http://es1.example.com:9200/_cluster/health -v
```

Listing Exports and Clients

Exports – To list active exports from the SwarmFS server, run the following command:

```
ganesha_mgr show_exports
```

```
Show exports
```

```
Timestamp: Thu May 17 17:38:54 2018 76205812 nsecs
```

```
Exports:
```

```
Id, path, nfsv3, mnt, nlm4, rquota, nfsv40, nfsv41, nfsv42, 9p, last
0, /, 0, 0, 0, 0, 0, 0, 0, 0, Thu May 17 17:38:02 2018, 488596707 nsecs
2, /nfsdatadirect, 0, 0, 0, 0, 0, 0, 0, 0, Thu May 17 17:38:02 2018, 488596707 nsecs
3, /filefly, 0, 0, 0, 0, 1, 0, 0, 0, Thu May 17 17:38:44 2018, 170782919 nsecs
```

Clients – To list active clients from the SwarmFS server, run the following command:

```
ganesha_mgr show_client
```

```
Show clients
```

```
Timestamp: Thu May 17 17:41:36 2018 780342324 nsecs
```

```
Clients:
```

```
IP addr, nfsv3, mnt, nlm4, rquota, nfsv40, nfsv41, nfsv42, 9p, last
::ffff:172.30.14.91, 0, 0, 0, 0, 1, 0, 0, 0, Thu May 17 17:38:44 2018 170782919 nsecs
```

Matching Requests between SwarmFS and Storage Logs

An implementation can have large numbers of unrelated parallel NFS requests. Enable verbose (DEBUG) logging and make use of these labels logs can be traced through if, for troubleshooting, storage requests need to be traced back to individual SwarmFS files being read and written:

- request-type prefix
- fileid
- download/upload id
- part number



Caution

Do not enable DEBUG logging any longer than necessary if exports are mounted directly on the SwarmFS server.

Missing Custom Header

It may be due to having an invalid name if an expected custom header is missing from an object. SwarmFS skips malformed custom headers silently.

See [Custom Metadata Headers](#) for the rules of custom header naming in Swarm Storage.

Users Lost Permissions

If after a few hours a user becomes unable to read or write files, despite having permissions, session authorization may need to be enabled in the SwarmFS exports.

To have normal reads, writes, and attribute updates go through session authorization, superuser access needs to be set up, which is necessary for numerous operations:

- Directory management (create, delete, rename)
- File renaming
- Certain :metadata writes

This is how to enable session-specific authorization for 2.1 and higher:

1. First, to create session authorization, configure token admin credentials in NFS (user + pass, or token).
2. Next, verify *one* of the following:
 - Specify a user with full access granted by the applicable policy in the **User Credentials** of the NFS export configuration.
 - Verify the token admin as full access granted by the applicable policy.

Performance Issues

See also **Optimizing Performance** in [SwarmFS Deployment](#).

Symptom	Actions
Gateway is overloaded and experiencing timeouts from excessive SwarmFS requests	Reconfigure the client (such as Samba) to use larger blocksizes (buffers) to transfer data, such as 1 MB or higher. (NFS-785)
Performance for larger files is lagging	<p>Increase the storage setting <code>ec.SegmentConsolidationFrequency</code> to 100. (NFS-786)</p> <p>Check whether the storage cluster is nearly full, and add capacity; <i>increasing this setting generates additional trapped space.</i></p>

SwarmFS Server Installation

- [System Requirements](#)
 - [Important](#)
- [Preparing Export Configurations](#)
- [Installing SwarmFS](#)
 - [Note](#)

System Requirements

These are minimum requirements for SwarmFS servers in production:

OS	RHEL/CentOS 7	CentOS 7.4+ SELinux Policy Management: Edit <code>/etc/selinux/config</code> and set <code>SELINUX=permissive</code> , or map the ports needed by Swarm using semanage-port : <pre>semanage port -a -t http_port_t -p tcp 91 semanage port -a -t http_port_t -p tcp 9200 semanage port -a -t http_port_t -p tcp 9300</pre>
CPU	2 cores	
RAM	4 GB (minimum)	4GB for a single export Add an extra 4GB for each additional export when using default export memory settings
Drive	Dedicated <code>/var/log</code> partition	Provide at least 30 GB for SwarmFS logs

The RAM requirements increase with the number of *exports*, but the greatest impact on CPU and RAM is driven by the *number of concurrent client operations*. The more concurrent client operations being served, the more RAM and CPU cores needed. How much and how many depends on what the clients are handling (in terms of file/object sizes), so focus on whether the allocated resources are being fully utilized. As they near full utilization, add more.



Important

- **VMs** – Where write performance is critical, install the SwarmFS server on physical hardware (not as a VM).
- **Paging** – Paging negatively impacts performance. If paging out does occur, increase RAM rather than increasing disk space for paging.
- **DEBUG** – When mounting exports directly on the SwarmFS server, do not enable verbose (DEBUG) logging.
- **Shared writes** – Linux UMASK (User file creation MASK) defaults to 0022, so directories are created with the permissions 755. The owner may write to the file. To share file writes among multiple users of the same group, change the permissions on the folder, or set the UMASK to 0002 so it applies to all newly created folders. Add it to `/etc/.bashrc` or `/etc/profile` to persist changes across reboots.

Preparing Export Configurations

Prepare the export configuration files needed to reference before installing any SwarmFS servers:

1. Select **Settings > NFS** in the Swarm UI.
2. Define one or more server groups (each group having one export configuration URL to be shared among a set of servers). See [SwarmFS Export Configuration](#).
3. Define one or more exports within each group, which become the mount points for applications.

i Tip

The export URL is non-functional but valid before defining exports, and exports can be added to and updated at any point.

Installing SwarmFS

The RPM for SwarmFS includes an interactive script for completing the needed SwarmFS configuration for a specific Swarm Storage cluster. The script prompts for the URL of the SwarmFS JSON export configuration created for Swarm Storage, so the NFS exports must be defined via the Swarm UI.

i Note

The script enables core file generation; it can be disabled via the `nfs-ganesha.service` file (`/usr/lib/systemd/system/`) or through the system-wide configuration.

Run the scripted process on each CentOS 7 system to be a SwarmFS server:

1. Download the SwarmFS package from the [Downloads section](#) on the [DataCore Support Portal](#).
2. Install the EPEL release, which has the needed packages for NFS:

```
yum -y install epel-release
```

3. Some later EPEL releases are missing the needed Ganesha and Ganesha utilities packages, so install those:

- a. Navigate to the NFS community build service: <https://cbs.centos.org/koji/buildinfo?buildID=10626>
- b. Scroll down to the RPMs list and download both packages:

- `nfs-ganesha-<version>.rpm`
- `nfs-ganesha-utils-<version>.rpm`

- c. Install both packages:

```
yum -y install nfs-ganesha-<version>.rpm
yum -y install nfs-ganesha-utils-<version>.rpm
```

4. Install the Swarm RPMs:

```
yum install caringo-nfs-libs-<version>.rpm
yum install caringo-nfs-<version>.rpm
```

5. Run the SwarmFS configuration shell script (which is in the path and located in `/usr/bin`). The script generates the local SwarmFS service configuration, validates the environment, enables the SwarmFS services, and then starts the SwarmFS services.

- a. Copy in the export configuration URL from the **NFS** tab of the Swarm UI:

```
/api/nfs/configurations/_plain1?
sptid=28d562ed002a34c54f2b0ccbf6f5476b88a23177e3b24c5333182d3143984e41
```

- b. Add the host and the user and password to be used for authentication:

```
SwarmNFS-config nfsadmin@password http://172.30.13.95:91/api/nfs/configurations/_plain
```

c. Run the script again if the SwarmFS-config script fail to complete (such as due to a network problem). It is configured to restart if the NFS service fails.

6. Enable the service to allow SwarmFS to start automatically on boot:

```
systemctl enable /usr/lib/systemd/system/nfs-ganesha.service
```

7. Run this command to verify the status of the services:

```
systemctl status nfs-ganesha
```

This status report is comprehensive and includes which processes are running.

 **Tip**

On startup, SwarmFS may generate WARN level messages about configuration file parameters. These are harmless and can be ignored. (SNFS-216)

Ganesha Operations for SwarmFS

These are Ganesha management operations that are helpful for use with SwarmFS:

- [Show Exports](#)
- [Show Clients](#)
- [Get Statistics](#)
- [Change Logging Level](#)
 - [Enable debug](#)
 - [Turn off debug](#)

Show Exports

`ganesha_mgr show_exports` – lists current SwarmFS export definitions:

```
[root@swarmnfs ~]# ganesha_mgr show_exports
Show exports
Timestamp: Wed Mar 13 09:24:36 2019 651073619 nsecs
Exports:
Id, path, nfsv3, mnt, nlm4, rquota, nfsv40, nfsv41, nfsv42, 9p, last
0, /, 0, 0, 0, 0, 0, 1, 0, 0, Wed Mar 13 09:20:41 2019, 939685320 nsecs
2, /nfsvaultdemo, 0, 0, 0, 0, 0, 1, 0, 0, Wed Mar 13 09:20:41 2019, 997373120 nsecs
```

Show Clients

`ganesha_mgr show_client` – lists current SwarmFS clients:

```
[root@swarmnfs ~]# ganesha_mgr show_client
Show clients
Timestamp: Wed Mar 13 09:25:20 2019 570496447 nsecs
Clients:
IP addr, nfsv3, mnt, nlm4, rquota, nfsv40, nfsv41, nfsv42, 9p, last
::ffff:127.0.0.1, 0, 0, 0, 0, 0, 1, 0, 0, Wed Mar 13 09:24:42 2019 327478281 nsecs
```

Get Statistics

`ganesha_stats fast` – reports global statistics for the current SwarmFS operation:

```
[root@swarmnfs ~]# ganesha_stats fast
Timestamp: Wed Mar 13 09:25:42 2019326319591 nsecs
Global ops:
NFSv3:

NFSv4:
ACCESS           :      15756
GETATTR         :      47299
GETFH           :         38
LOOKUP          :         37
PUTFH           :     47303
PUTROOTFH       :         2
REaddir         :         5
EXCHANGE_ID     :         1
CREATE_SESSION  :         1
SECINFO_NO_NAME :         1
SEQUENCE        :     157421
RECLAIM_COMPLETE :         1
```

Change Logging Level

`ganesha_mgr set_log` – allows dynamic changes to the logging level.

Enable debug

```
ganesha_mgr set_log COMPONENT_FSAL FULL_DEBUG
```

Turn off debug

```
ganesha_mgr set_log COMPONENT_FSAL NIV_EVENT
```

SwarmFS Listings

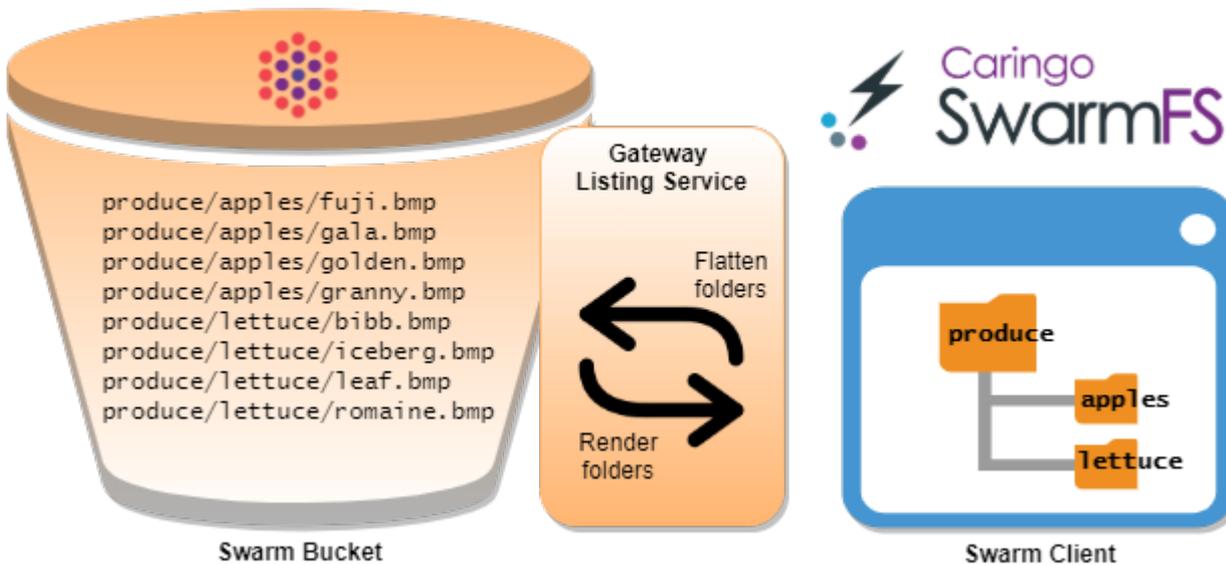
The defined SwarmFS exports allow users access the named objects in Swarm storage. The Swarm namespace is flat below the bucket level. Users need to see and interact with these objects in a familiar way. Content Gateway includes a *listing service* translating the object store in to virtual file hierarchies. (v3.0)

Gateway achieves a file-system-like hierarchical structure on those objects by splitting them in to “files” and “folders,” where

- **files** are content objects (such as videos)
- **folders** are file system directories Gateway renders from object names with a prefix with one or more delimiters

Folders come in two types:

- **Synthetic folders** do not exist in Swarm: they are client-side creations from the names of objects in Swarm.
- **Pseudo folders** are stored in Swarm as a object with no content, whose name ends in the delimiter (“/”). These objects define any placeholder folders as created by users (`reserve/this/folder/`).



Listing Behaviors

The listing service translates any prefix+delimiter it finds in an object name in to a folder listing. It renders data so it exactly matches the object stored in Swarm, other than a few milliseconds to acquire it. Listings sort by name, as is the S3 standard.

Delimiters – It uses a globally configured delimiter (defaults to slash “/”) in Swarm (the storage setting `search.pathDelimiter`); different characters can be used but performance may be slower (see [Settings Reference](#)). SwarmFS converts slashes to backslashes inline during listing transfers if a client *expects* backslashes to be the directory delimiter. SwarmFS always converts backslashes to forward slashes when communicating with Swarm.

i Important

Object and bucket names must not *start* with a delimiter (/) to be valid. They do not appear in SwarmFS listings if existing objects are named like this. Rename and upload objects with valid names if this occurs.

Metadata – Listing returns objects with all metadata. It stores and renders pseudo folders as objects, which may hold usable folder and POSIX metadata. See [Metadata Headers](#).

Listing delay – New objects may be missed, or existing objects show up in the results twice, temporarily if native Swarm objects change *between* fetches. Objects created in Swarm natively (not via SwarmFS) can take up to 5 minutes plus the [Search Feed's Batch Timeout](#) to appear in SwarmFS listings, because they must be indexed by Elasticsearch. For best listing performance, lower the search feed's **Batch Timeout** to 1 or 0 (recommended). See [Search Feeds](#).

Versions – For versioned buckets, SwarmFS provides all object versions in the listings, or, on client request, returns the current versions and delete markers for the deleted objects. See [Implementing Versioning](#).

Encoding – SwarmFS supports UTF-8 encoded names, and the REST interface returns UTF-8-encoded results.

Exclusive opens – SwarmFS supports exclusive opens of a file (O_EXCL and O_CREATE) but does not support exclusive reopens (EXCLUSIVE4).



Note

The listing service does not currently include unnamed objects, caching, folder locking/leasing, or client notification of namespace changes.

How Listings Appear to Users

SwarmFS creates a simulation: it translates each forward slash (/) in any object name in to a traditional directory delimiter to present named object listings as if they are in a traditional file system. SwarmFS then presents a view of objects at the simulated directory level.

Suppose these named objects exist in the bucket "AcmeBucket":

```
Sales/Leads/campaigns/2019/Jan/list.xls
Sales/Leads/campaigns/2019/Feb/list.xls
Sales/Leads/campaigns/addword.xls
Sales/Leads/campaigns/partners.xls
```

- By default, when a user requests a listing of the bucket contents, SwarmFS provides a simulated view of the *first* level of contents within the bucket.
- SwarmFS returns a new listing for a directory each time a user opens a specific directory level *within* the bucket (such as `Sales/Leads/campaigns/`).
- A single file is returned if a user requests a listing for a file (such as `Sales/Leads/campaigns/addword.xls`).
- SwarmFS creates a new empty object with Directory attribute metadata, which marks it as a pseudo directory if a user creates a new directory (such as `AcmeBucket/Sales/Leads/campaigns/2019/Mar/`).
- SwarmFS removes it and all files it contains if a user deletes a directory.
- A user cannot *move or rename* a directory. Instead, they need to create the directory they want and move the files into it. (NFS-607)

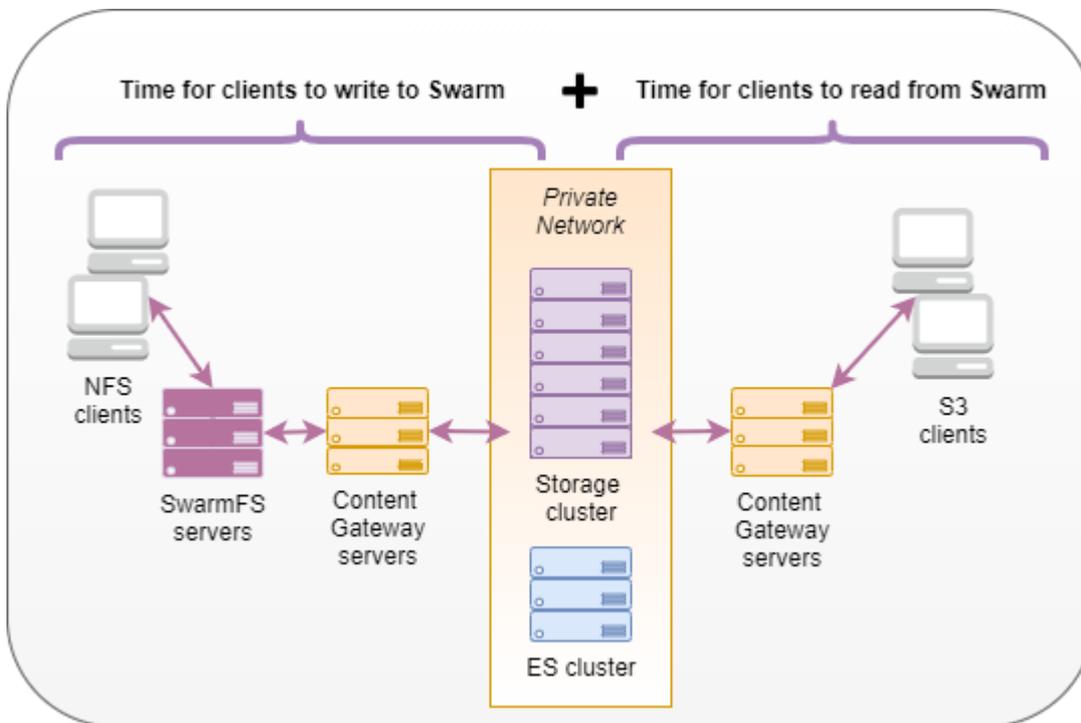
Simulated directory objects exist to support the expectations of end users and applications: objects can be written to a new virtual directory by including it in the pathname.

SwarmFS Planning

- [Planning the SwarmFS Environment](#)

SwarmFS can coexist with other applications running on the same Linux server, although SwarmFS expects to have sole ownership of its assigned ports and resources. SwarmFS can be deployed onto the same Linux server as the Content Gateway, or it can run on its own dedicated operating system instance. Note the following for your planning:

- **Stateless** – SwarmFS is stateless, so each process stores no run-time configuration outside of volatile RAM locally, with the exception of a basic configuration file read on startup (see [SwarmFS Export Configuration](#)). Should a process fail or restart, any incomplete operations that it was processing are lost, and it restarts clean and stateless.
- **Multiple active** – There is no limit to the number of SwarmFS servers that can be online at any time: multiple SwarmFS processes can be active running on different servers simultaneously. SwarmFS running on different servers can be configured identically and present the same object view as other active running SwarmFS. SwarmFS servers can be added and removed independently of the others, and any single SwarmFS server going offline has no effect on other running instances.
- **Performance** – The job of an NFS server is to keep data and metadata well stored and to move it efficiently; the job of an NFS client is to translate and adapt the NFS protocol to the local environment efficiently, which is more challenging. Client performance is complex, driven by the performance of servers and drives, as well as the efficiency of networking, caching, and data structures. Many areas affect the throughput achievable, from writing data using an NFS client through the ingest in Swarm Storage to the ability to be read by a Swarm client:



Planning the SwarmFS Environment

Component	Value	Notes
Is the Swarm cluster already running?	yes no	Swarm must be running before SwarmFS can be configured
Is a Elasticsearch cluster installed and running?	yes no	Elasticsearch must be running before SwarmFS can be configured

Is there a working Swarm “Search Full Metadata” feed to the Elasticsearch cluster?	yes no	SwarmFS requires a full metadata search feed to be running
Is Content Gateway running?	yes no	Content Gateway must be running before SwarmFS can be configured
What are the addresses for Gateway?		Make note of the IP addresses or DNS-resolvable names for your Gateways (one or more)
Is port 91 (default) open between SwarmFS and Gateway?	yes no	SwarmFS must be able to connect to the Swarm management API, either directly to swarm or via the gateway proxy
How many SwarmFS servers are installed?		You can have one or more SwarmFS Servers
Are all SwarmFS servers present the same NFS exports?	yes no	SwarmFS servers can present the same exports, or different exports, you can also have groups of SwarmFS servers presenting different exports
Which operating system is SwarmFS installed on?	CentOS RHEL	If using RHEL (Red Hat Enterprise Linux), your site needs a Red Hat license
Is SELinux enabled and enforcing on the SwarmFS servers?	yes no	<p>If SELinux is enabled, it must allow SwarmFS to open these network connections:</p> <ul style="list-style-type: none"> • Elasticsearch (default port 9200) • Swarm Management API (default port 91)

Migrating from Traditional Storage

- [Advantages of Object Storage](#)
 - [Never-ending storage systems](#)
 - [Bullet-proof protection](#)
 - [Rich metadata](#)
- [Advantages of Deploying Content Gateway](#)
 - [Tenants, Domains, and Buckets](#)
 - [Organizing by Tenant](#)
- [Migration Planning](#)
 - [Adapting the Legacy Structure](#)
 - [Best Practices for Restructuring](#)
 - [Planning Areas](#)

Advantages of Object Storage

Object storage brings capabilities making aspects of traditional file systems obsolete.

- It works as a unified, self-scaling, self-protecting, and self-healing pool of storage requiring no backup (and may even be too large to back up in a traditional way).
- It offers enhanced metadata (*data about data*), which can be customized and leveraged programmatically.
- It includes large-scale, high-performance searching based on rich metadata.

Never-ending storage systems

In the end, all traditional file systems and storage systems run up against hard limits. Whether it is at the volume/block layer level or at the partition level, an upper limit is always faced on how large a LUN (logical unit number) can be made or where a partition begins to become unmanageable, due to size. Object storage offers an effectively limitless namespace and storage layer to house growing data.

Large data LUNs are created by aggregating multiple disks together using hardware or software RAID technologies and accessing them using a fast interconnect, like fiber or iSCSI. These RAID volumes have limits and durability characteristics, which creates challenges for LUN sizing. Different SAN manufacturers have different limits on the ideal sizes and distribution of LUNs. IT administrators must prioritize the type of data protection level and speed each time they commission new storage. It is rarely a matter of making the largest volume possible and offering it out to users to carve up as they like, so dynamically scaling these systems is challenging, if not impossible.

In contrast, Swarm clusters are unified volumes of storage with the ability to share a single protection profile or apply different protection profiles within the cluster. Add new hardware to the cluster and allow Swarm to scale.

Bullet-proof protection

Data loss at a small scale, such as 1 or 2 disk failures for a single RAID volume in a SAN or local RAID group, is survivable: replace those failed disks and suffer decreased performance while the parity is rebuilt after a period of hours or days, depending on the size of the volume and the amount of data on it. Multi-disk failures are common enough, and hard disk capacities increase with time, leading to longer rebuild times and larger datasets and backup times.

Swarm object storage is inherently designed to sustain and heal from multiple disk failures and, depending on the configuration, *multiple server failures*. In addition to content protection policies allowing precise cluster, domain, and bucket-level controls, Swarm also offers additional layers of data protection implemented to support an organization's needs for protection:

- [Replication on Write](#), for immediate backup of ingested content
- [Mirrored clusters](#), using remote replication
- [Object-level versioning](#) (S3-compatible)
- [DR \(disaster recovery\)](#) via a feed to S3 cloud storage

Rich metadata

Information *about* the data is now as important as the data itself, for analytics, retrieval, and value-add processes. With a traditional file system, such as NTFS or ext4, metadata for the file is fixed by the file system and is limited to system-side information (access times, owner, attributes). With Swarm, up to 32 KB of custom metadata can be stored with *each* object, which is a tremendous amount of text-based information.

A growing number of specialty file formats have emerged to allow files to embed critical information about *the contents* like a passport, such as the richness of data modern digital cameras store with each photo, capturing the location, camera make and model, resolution, speed, exposure, and more. In most of these cases, the file contains the metadata itself, and the application used to view the file restricts what metadata is visible to the user.

In the same way, extended metadata becomes *part* of each object being stored in Swarm and so cannot be lost. In a Swarm cluster, all metadata associated with a file is stored as header information on the file itself. This header information is viewed using a HTTP HEAD of the file, requiring no special drivers or applications.

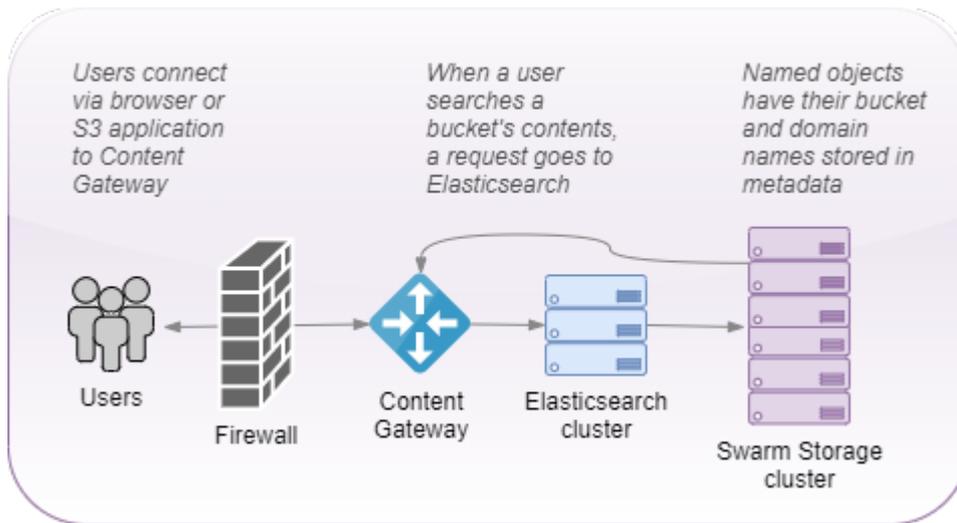
In addition, Swarm allows creating and storing [standalone metadata annotations](#) as a header-only object associated with an existing content object. The ability to keep extending custom metadata and add metadata to read-only objects is effectively unlimited.

Advantages of Deploying Content Gateway

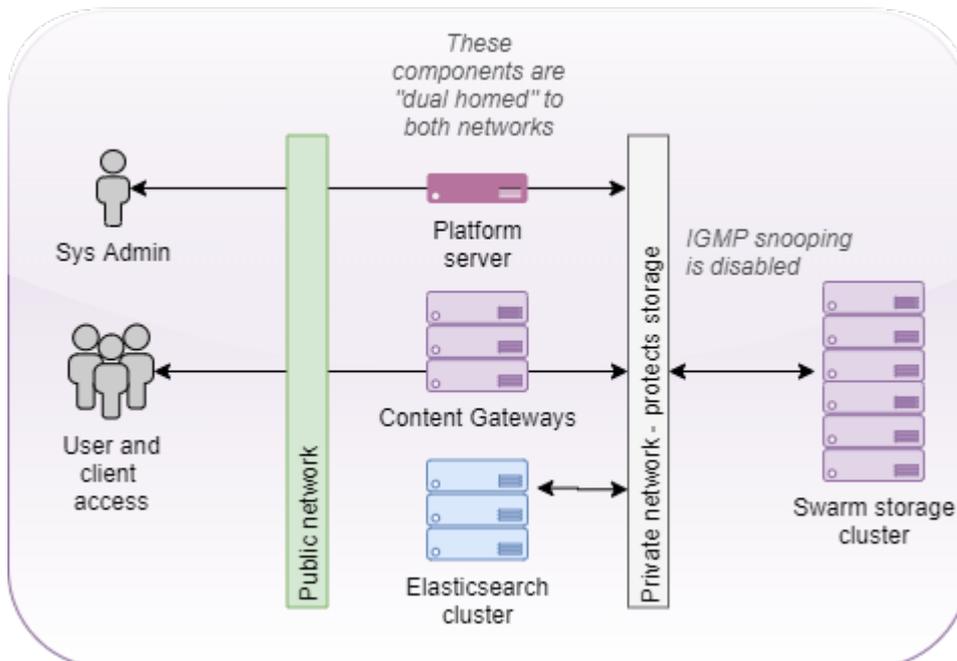
Implementing Swarm with Content Gateway provides an organization with authentication, a browser UI for end users, S3 protocol access, and enhanced multi-tenancy. Multi-tenancy (discussed below) can be a critical tool for dividing and delegating content access and structure within large organizations.

Below is a basic Swarm deployment leveraging Content Gateway:

- A 6-chassis **Swarm cluster**, for hardware resilience
- **Elasticsearch** cluster for dynamic searching
- **Content Gateway**

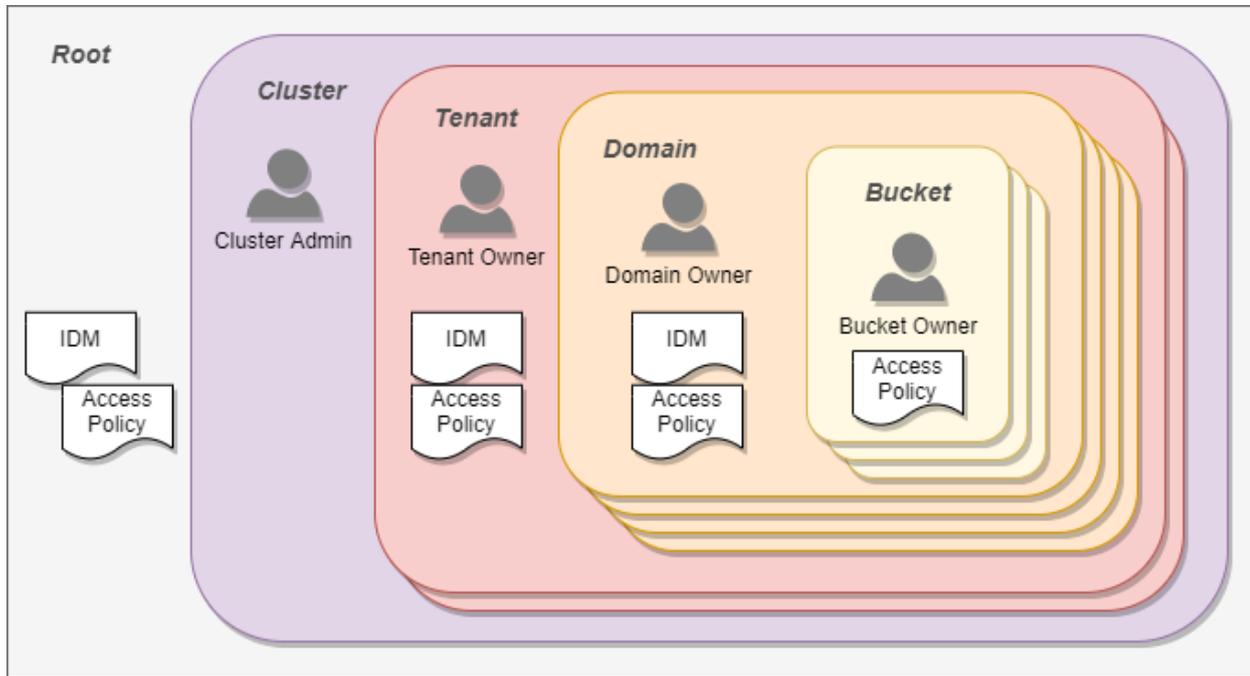


The Swarm Storage cluster is protected within a dedicated private network, and all client and application traffic passes through Content Gateway:



Tenants, Domains, and Buckets

Swarm offers multiple levels of access. The following focuses on tenants, domains, and buckets:



- **Tenant** – A *tenant* is a hierarchy owning one or more storage domains. Each tenant scope can define a separate identity management system so users and groups within them are separated from those in other tenants. The tenant administrators have the ability to create and access storage domains on behalf of the tenant, and they can delegate management duties for the storage domains they create. The tenant scope does not store end-user data; it is a meta store for information about the tenant, users, and storage domains.
- **Domain** – The *domain* scope is directly tied to a Swarm storage domain and is where end-user data is kept. The SCSP and S3 storage protocols create and use data within the domain scope. While the domain scope can inherit user and group identity information from the tenant, it also has the ability to define a separate identity management system. The domain administrators can create and access all content within the storage domain. They can optionally delegate control of storage buckets to individual users or groups.
- **Bucket** – The *bucket* scope is directly tied to a bucket existing within the Swarm storage domain. While access control policies can be defined for every bucket, there is no option for an identity management system definition at the bucket scope. All buckets with a domain share the domain's identity management system definition.

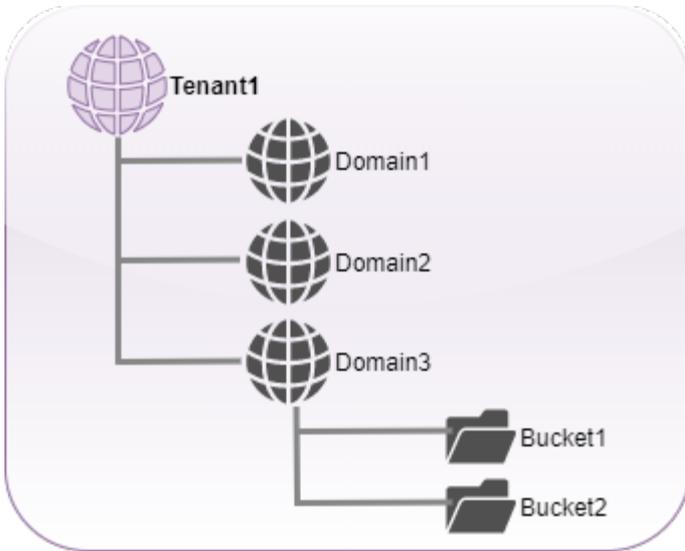
In short: *A tenant holds multiple domains, and a domain holds multiple buckets.*

Organizing by Tenant

Outside of multi-tenancy environments, tenants are useful for grouping similar storage areas in a cluster.

Single tenant, wildcard DNS

Here is a top-level structure:



Tenant1's auth and protection levels are inherited by the domains lower down. **Domain3** has buckets (represented here by folders).

Note

Even though the tenant is a special type of domain specific to Gateway, it is still a domain.

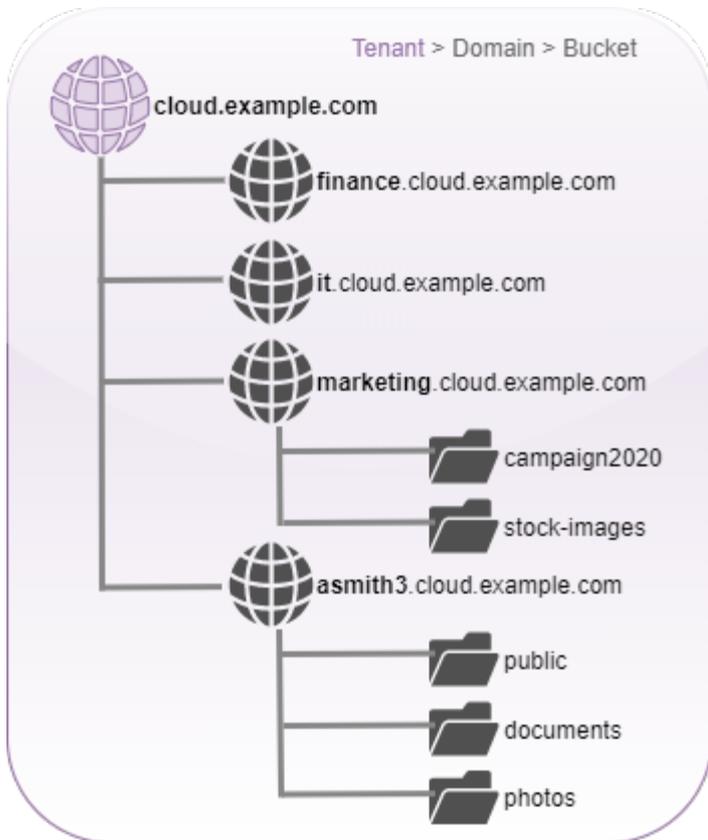
Each of these domains can also be fully qualified within the corporate DNS structure.

Tip

Use wildcards so there is no need to add DNS records for every new domain as they are created. This allows users to create separate domains, and DNS resolution happens automatically as long as the domains are created with a similar naming structure.

One domain per department, employee

Create a wildcard DNS record for the gateway's address: `*.cloud.example.com`

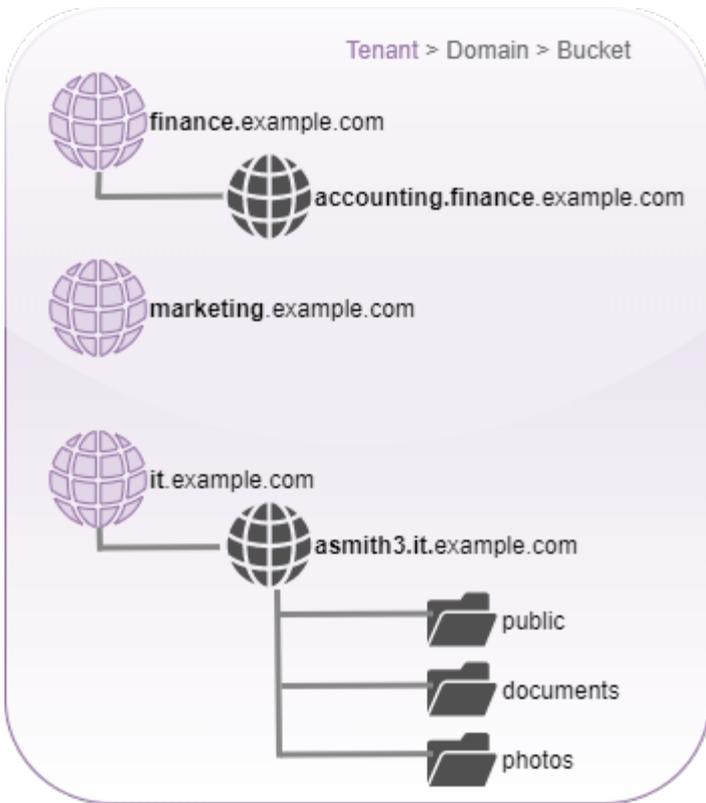


Each domain created here represents a single department in the organization. A domain can be created for every employee as there is no limit to the number of domains within the storage cluster. The last domain is an employee domain: `asmith3.cloud.example.com`

Employees can create as many buckets as they wish *within* separate domains, to further subdivide content.

One tenant per division

It may make sense to have more than one top-level tenant for an organization. Provide each corporate division a separate tenant so it can create and control separate departmental and employee domains. This provides an additional level of organization and authorization to work with.



Create the most readable and shortest path to the information relevant to users.

Verify the division is correct as it appears from a browser. The following URL is easy to interpret and access:

```
http://accounting.finance.example.com/fiscalresults2017/data.xls
http://<dept>.<division>.<org>.com/<bucket>/<filename>
```

Migration Planning

The following are guidelines to facilitate a smooth migration to object storage.

Adapting the Legacy Structure

When migrating data from a traditional block storage or file-sharing solution, carefully evaluate the structure as it already exists and decide how much of the structure to take forward.

Users do not like change. When a file server is added to an organization, tribal knowledge tends to develop and ingrained about what data goes where in an enterprise. New users are given access to the “P” drive or the “docs” folder, and, bit by bit, they learn where things are and where to put things. When implementing a new structure or a new file server, ask a user how they prefer the new system look. They may insist “Exactly like the one now!” There is no easy answer to combat this and it is a real challenge.

When performing a migration of any kind, *this* is the time to start to manage change in the organization, to minimize problems and resistance. It is also important to evaluate the old structure for duplication and dead wood early on, to eliminate it as part of the migration.

Best Practices for Restructuring

The following are lessons emerged from many implementations:

Do not bulk move folders to pseudo folders

- An object store offers immense flexibility; bulk moving folders of files removes the flexibility and keeps the older structures. This adds challenges to changes going forward.
- Any 1-to-1 movement needs to use pseudo folders, which are prefixes to an object name. Pseudo folders add challenges to object searches.
- Permissions and user attributes apply to the object, not the folder. Users can be disappointed if creating a pseudo folder thinking they share it and all files in it.

Convert pathnames using domains and buckets

- Think about what looks best in an object context if a very long pathname exists such as /year/month/day/filename. The shortest path is to have the domain as the year, with the bucket being a month+day context. For example: `2017-hq-videos.example.com/Sep-13/videofile.mp4`
- There is no need to have a date on a bucket name if the date is in the filename.

Use domains for data groups

- Provide separate domains if a large amount of similar data or data always used in the same workflow exists.

Use tenants/domains for applications

- Provide an application a separate tenant or domain if an organization uses a particular application whose dedicated data is used via the application.

Optimize for searches

- [Collections](#) are saved searches where the scope of the search can be the entire domain or a specific bucket.
- When creating domains and buckets, avoid creating a structure too granular for large searches. For example: Creating a bucket per hour may be excessive unless there is a lot in each bucket if creating a bucket per day in a domain for a certain type of data.

Planning Areas

Any migration project requires consultation with DataCore and planning around these key area. This requires all integration points in an environment be listed and diagrammed:

Namespace	<p>Strategy for mapping file systems to objects (discussed above)</p> <p>What FQDN (fully qualified domain name) and DNS setup to use for Gateway (see Content Gateway Implementation)</p>
Networking	<p>Work out, down to each port (see Setting up the Swarm Network), how all Swarm components integrate, to surface design issues</p> <p>List required applications and verify they can access storage regardless of network segment</p> <p>Evaluate need for HTTP versus HTTPS (see also Replicating Feeds over Untrusted Networks)</p> <p>Whether to use front-end load balancing or round robin</p>
Authentication	<p>Is LDAP or Active Directory integration being used?</p> <p>How does the current ACL structure map to Gateway ACLs? (see Content Gateway Authentication and Setting Tokens)</p>
Swarm clients (optional)	<p>SwarmFS Implementation</p> <ul style="list-style-type: none"> • Check minimum requirements if deployed client-side • Networking implications (Elasticsearch access and IP whitelisting) <p>FileFly</p>

Elasticsearch Implementation

Swarm integrates Elasticsearch and extends the Swarm API with commands for querying Swarm objects in terms of metadata. Through this feature, Swarm indexes object metadata in near real time and allows performing ad hoc searches (via [query commands](#)) on the attributes and metadata of your stored objects.

Swarm uses Elasticsearch servers for its metadata searching operations. You can deploy these servers for high-availability and horizontal scaling. Although high availability of the *search* cluster is not needed for high availability of the *storage* cluster, you may need it to service third-party analytics applications.

Important

For production-level responsiveness and redundancy, deploy at least *three* search servers. Follow the [Hardware Requirements for Elasticsearch](#).

You can return the results as JSON or XML, which you can import into your third-party analytics applications.

See also these sections:

- [Swarm Storage Release Notes](#)
- [Elasticsearch for Swarm](#) (configuration and administration)
- [Swarm Historical Metrics](#)
- [Storage SCSP Development](#)

Search components

The search infrastructure includes these components:

- **Swarm Storage cluster**, which is connected to the Elasticsearch servers through a Search Feed.
- **Search feed(s)**, which transmit the metadata from the storage cluster. Feeds iterate over data on storage nodes and use intermittent channel connections to distribute data to one or more configured destinations, including metadata search servers. See [Managing Feeds](#).

Tip

Because Swarm uniquely names each search feed index, you can configure additional feeds that use the same Elasticsearch cluster; plan for doubling or tripling the space demands on that server.

- **Elasticsearch servers**, which index the metadata and service search requests. This metadata can be reconstructed from the storage cluster, if needed.
- **Metrics curator service**, which can be installed on one of the Elasticsearch servers, or another system running RHEL/CentOS 7.
- **Client applications**, which access the Swarm cluster through SCSP commands.

Best practice

Devote the search cluster to Swarm-only usage, and do not store non-Swarm data in your search installation.

- [Configuring Elasticsearch](#)
- [Preparing the Search Cluster](#)
- [Installing Elasticsearch](#)
- [Hardware Requirements for Elasticsearch](#)
- [Scaling Elasticsearch](#)
- [Upgrading Elasticsearch](#)
- [Migrating from Older Elasticsearch](#)

Configuring Elasticsearch

Elasticsearch requires configuration and settings file changes to be made consistently across the Elasticsearch cluster.

- [Scripted Configuration](#)
- [Customization](#)
 - [Elasticsearch Config File](#)
 - [Systemd \(RHEL/CentOS 7\)](#)
 - [Environment Settings](#)
 - [JVM Options](#)
 - [Log Setup](#)

Scripted Configuration

Using the provided configuration script automates in-place Elasticsearch upgrades as well as the essential configuration that Elasticsearch requires for use with Swarm.

The script handles the following:

- Upgrading Elasticsearch in place (using the same index) if it detects a supported version (6.8.6) is already installed and configured
- Editing `/etc/elasticsearch/elasticsearch.yml` (except for changing the `path.data` variable to use a different data directory)
- Editing `/etc/elasticsearch/log4j2.properties`
- Editing `/usr/lib/systemd/system/elasticsearch.service`
- Editing `/etc/sysconfig/elasticsearch`
- Creating the override file for Systemd: `/etc/systemd/system/elasticsearch.service.d/override.conf`

Bulk Usage

This method is most efficient for a large number of nodes and/or have manual configurations to apply to the `elasticsearch.yml` (see next section).

1. On the first Elasticsearch node, run the configuration script provided in `/usr/share/caringo-elasticsearch-search/bin/`. This script prompts for the needed values as it progresses:

```
/usr/share/caringo-elasticsearch-search/bin/configure_elasticsearch_with_swarm_search.
```

2. The script generates custom configuration files for each of the nodes in the Elasticsearch cluster. (v10.x)

- The current node's file is `/etc/elasticsearch/elasticsearch.yml`.
- The other nodes' files (if any) are `/etc/elasticsearch/elasticsearch.yml.<node-name-or-ip>`

3. Follow the **Customization** details (below) to update the YAML files further, such as to change Elasticsearch's `path.data` (data directory).

Logging

- Update log files to match your data path or other customizations.
- Update the rollingfile appender to delete rotated logs archives, to prevent running out of space.

4. For the next and all remaining nodes, complete these steps:

- a. On the next Elasticsearch node, copy over the appropriate file as `/tmp/elasticsearch.yml.esnode8`.
- b. With the YAML file in place, run the configuration script with the `-c` argument, so it uses the existing file.

```
configure_elasticsearch_with_swarm_search.py -c \  
/tmp/elasticsearch.yml.esnode8
```

- c. Move to the next node, if any.

5. Resume the installation to turn on the service: [Installing Elasticsearch](#) or [Migrating from Older Elasticsearch](#)

Non-Bulk Usage

1. On the first Elasticsearch node, run the configuration script provided in `/usr/share/caringo-elasticsearch-search/bin/`. This script prompts for the needed values as it progresses:

```
configure_elasticsearch_with_swarm_search.py
```

2. The script generates a custom `/etc/elasticsearch/elasticsearch.yml` configuration file for the current node as well as ones for each of the nodes, which can be ignored. (v10.x)
3. Following the **Customization** details below to update the YAML file further, such as to change Elasticsearch's `path.data` (data directory).

Logging

- Update log files to match your data path or other customizations.
- Update the rollingfile appender to delete rotated logs archives, to prevent running out of space.

4. Run the script the same way on each remaining ES node, answering the prompts consistently and reapplying any manual configurations.
5. Resume the installation to turn on the service: [Installing Elasticsearch](#) or [Migrating from Older Elasticsearch](#)

Customization

- [Elasticsearch Config File](#)
- [Systemd \(RHEL/CentOS 7\)](#)
- [Environment Settings](#)
- [JVM Options](#)
- [Log Setup](#)

The paths given are relative to the Elasticsearch installation directory, which is assumed to be the working directory.

Caution

- Errors in adding and completing these settings can prevent the Elasticsearch service from working properly.
- If Elasticsearch's `path.data` location is customized from the default, adjust all references to it below to reflect the new location.

Elasticsearch Config File

Version differences

The Elasticsearch configuration settings have changed with each major release. To track how they changed since Elasticsearch 2.3.3, see [Elasticsearch Configuration Differences](#).

Edit the Elasticsearch config file: `/etc/elasticsearch/elasticsearch.yml`

<code>action.auto_create_index: "+csmeter*,+*_nfsconnector, watches, .triggered_watches,.watcher- history-*</code>	Needed to disable automatic index creation, csmeter indices, and Swarm NFS connectors. (v10.1)
<code>cluster.name: <ES_cluster_name></code>	Provide the Elasticsearch cluster a unique name, which is unrelated to the Swarm cluster name. <i>Do not use periods in the name.</i> <div data-bbox="419 1497 1481 1703" style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <h3> Important</h3> <p>To prevent merging, it <i>must</i> differ from the <code>cluster.name</code> of the legacy ES cluster, if one is operating.</p> </div>
<code>node.name: <ES_node_name></code>	Optional. Elasticsearch supplies a node name if one is not set. <i>Do not use periods in the name.</i>
<code>network.host: _site_</code>	Assign a specific hostname or IP address, which requires clients to access the ES server using that address. If using a hostname, update <code>/etc/hosts</code> . Defaults to the special value, <code>_site_</code> .

cluster.initial_master_nodes	(ES 7+) For first-time bootstrapping of a production ES cluster. Set to an array or comma-delimited list of the hostnames of the master-eligible ES nodes whose votes should be counted in the very first election.
discovery.zen.minimum_master_nodes: 3	(ES 6 only) Set to (number of master-eligible nodes / 2, rounded down) + 1. Prevents split-brain scenarios by setting the minimum number of ES nodes online before deciding on electing a new master.
discovery.seed_hosts	(ES 7+) Enables auto-clustering of ES nodes across hosts. Set to an array or comma-delimited list of the addresses of all master-eligible nodes in the cluster.
discovery.zen.ping.unicast.hosts: ["es0", "es1"]	(ES 6 only) Set to the list of node names/IPs in the cluster, verifying all ES servers are included. Multicast is disabled by default.
gateway.expected_nodes: 4	Add and set to the number of nodes in the ES cluster. Recovery of local shards starts as soon as this number of nodes have joined the cluster. It falls back to the <code>recover_after_nodes</code> value after 5 minutes. This example is for a 4-node cluster.
gateway.recover_after_nodes: 2	Set to the minimum number of ES nodes started before going into operation status: <ul style="list-style-type: none"> • If total nodes is 1 or 2, set to 1. • If total nodes is 3 or 4, set to 2. • If total nodes is 5 to 7, set to the number - 2. • If total nodes 8 or more, set to the number - 3.
bootstrap.memory_lock: true	Set to lock the memory on startup to verify Elasticsearch does not swap (swapping leads to poor performance). Verify enough system memory resources are available for <i>all</i> processes running on the server. To allow the <code>elasticsearch</code> user to disable swapping and to increase the number of open file descriptors, the RPM installer makes these edits to <code>/etc/security/limits.d/10-caringo-elasticsearch.conf</code> : <pre># Custom for Caringo Swarm elasticsearch soft nofile 65536 elasticsearch hard nofile 65536 elasticsearch soft nproc 4096 elasticsearch hard nproc 4096 # allow user 'elasticsearch' memlock elasticsearch soft memlock unlimited elasticsearch hard memlock unlimited</pre>
path.data: <path_to_data_directory>	By default, <code>path.data</code> goes to <code>/var/lib/elasticsearch</code> with the needed ownership. Choose a separate, dedicated partition of ample size, and make the <code>elasticsearch</code> user the owner of that directory to move the Elasticsearch data directory: <pre>chown -R elasticsearch:elasticsearch <path_to_data_directory></pre>
thread_pool.write.queue_size	The size of the queue used for bulk indexing. This variable was called <code>threadpool.bulk.queue_size</code> in earlier Elasticsearch versions.

Systemd (RHEL/CentOS 7)

Create a systemd override file for the Elasticsearch service to set the `LimitMEMLOCK` property to be unlimited.

1. Create the override file:

```
/etc/systemd/system/elasticsearch.service.d/override.conf
```

2. Add this content:

```
[Service]
LimitMEMLOCK=infinity
```

3. Load the override file; otherwise, the setting does not take effect until the next reboot:

```
sudo systemctl daemon-reload
```

Environment Settings

Edit the environmental settings: `/etc/sysconfig/elasticsearch`

<code>MAX_OPEN_FILES</code>	Set to 65536
<code>MAX_LOCKED_MEMORY</code>	Set to unlimited (prevents swapping)

JVM Options

Edit the JVM settings to manage memory and space usage: `/etc/elasticsearch/jvm.options`

<code>-Xms</code>	Set to half the available memory, but not more than 31 GB.
<code>-Xmx</code>	Set to half the available memory, but not more than 31 GB.

GC logs (optional) – By default, Elasticsearch enables GC logs. These are configured in `jvm.options` and output to the same default location as the Elasticsearch logs. The default configuration rotates the logs every 64 MB and can consume up to 2 GB of disk space. Disable these logs until needed to troubleshoot memory leaks. To disable them, comment out these lines:

```
#8:-Xloggc:/var/log/elasticsearch/gc.log
#8:-XX:+UseGCLogFileRotation
#8:-XX:NumberOfGCLogFiles=32
#8:-XX:GCLogFileSize=64m
#9:-Xlog:gc*,gc+age=trace,safepoint:file=/var/log/elasticsearch/gc.log:utctime,pid,tags:filecount
```

Log Setup

To customize the logging format and behavior, adjust its configuration file: `/etc/elasticsearch/log4j2.properties`

In its default location, logging has the needed ownership. Choose a separate, dedicated partition of ample size, and make the `elasticsearch` user the owner of that directory to move the log directory:

```
chown -R elasticsearch:elasticsearch <path_to_log_directory>
```

Deprecation log

This is the log of deprecated actions, to inform for future migrations. Adjust the log size and log file count for the deprecation log:

Update to these values

```
appender.deprecation_rolling.policies.size.size = 2097152
appender.deprecation_rolling.strategy.max = 25
```

By default, deprecation logging is enabled at the WARN level, the level at which *all* deprecation log messages are emitted. To avoid having large warning logs, change the log level to ERROR:

Change level

```
logger.deprecation.level = error
```

Elasticsearch Configuration Differences

The Elasticsearch configuration settings have changed with each major release; the following matrix shows how they changed by version.

Note
The "x" cells indicate that the setting is no longer used.

- [Elasticsearch Settings](#)
- [Memory Management](#)

Elasticsearch Settings

Location: /etc/elasticsearch/elasticsearch.yml

Elasticsearch 2.3.3	5.6.12	6.8.6	7.5.2
index.max_result_window	x	x	x
index.translog.sync_interval	x	x	x
index.translog.durability	x	x	x
bootstrap.mlockall	bootstrap.memory_lock		
threadpool.bulk.queue_size	thread_pool.write.queue_size		
script.inline script.indexed	script.inline	x	x
discovery.zen.ping.unicast.hosts	discovery.seed_hosts		
discovery.zen.minimum_master_nodes	cluster.initial_master_nodes		

Memory Management

Location	Setting	Elasticsearch 2.3.3	5.6.12	6.8.6	7.5.2
/etc/sysconfig/elasticsearch	ES_HEAP_SIZE	Set to half the physical memory on the machine, but not more than 31 GB.	x		
/etc/elasticsearch/jvm.options	-Xms -Xmx	x			Set to half the available memory, but not more than 31 GB.

Preparing the Search Cluster

Perform the following steps to prepare the search servers for Elasticsearch.

1. Verify the servers against the [Hardware Requirements for Elasticsearch](#).
2. Appropriately cable the servers to your network infrastructure so they are reachable from the Swarm nodes.
3. Install RHEL/CentOS 7 Linux and apply any required updates. Contact DataCore Support for questions about enabling or disabling IPv6.

Best practice

Use RHEL/CentOS Minimal Server (with Compatibility Libraries), which is the standard for Swarm development and testing. RHEL /CentOS Desktop consumes extra resources that Elasticsearch can use, alters the OS configuration to emphasize user interface vs. server performance, and requires additional updating and security maintenance.

4. Configure the servers with static IP addresses.
5. Configure DNS, if desired.
6. Adjust the server firewall rules. See <https://firewalld.org>.
Adjust the rules to permit the following ports if on CentOS 7 [install and run iptables](#):

- Allow public access on these ports:

```
firewall-cmd --permanent --zone=public --add-port=9200/tcp
firewall-cmd --permanent --zone=public --add-port=9300/tcp
firewall-cmd --reload
```

- Search the Support portal for *SwarmNFS 2.x - Access to Elasticsearch (IPTables)* for SwarmFS access.

Important

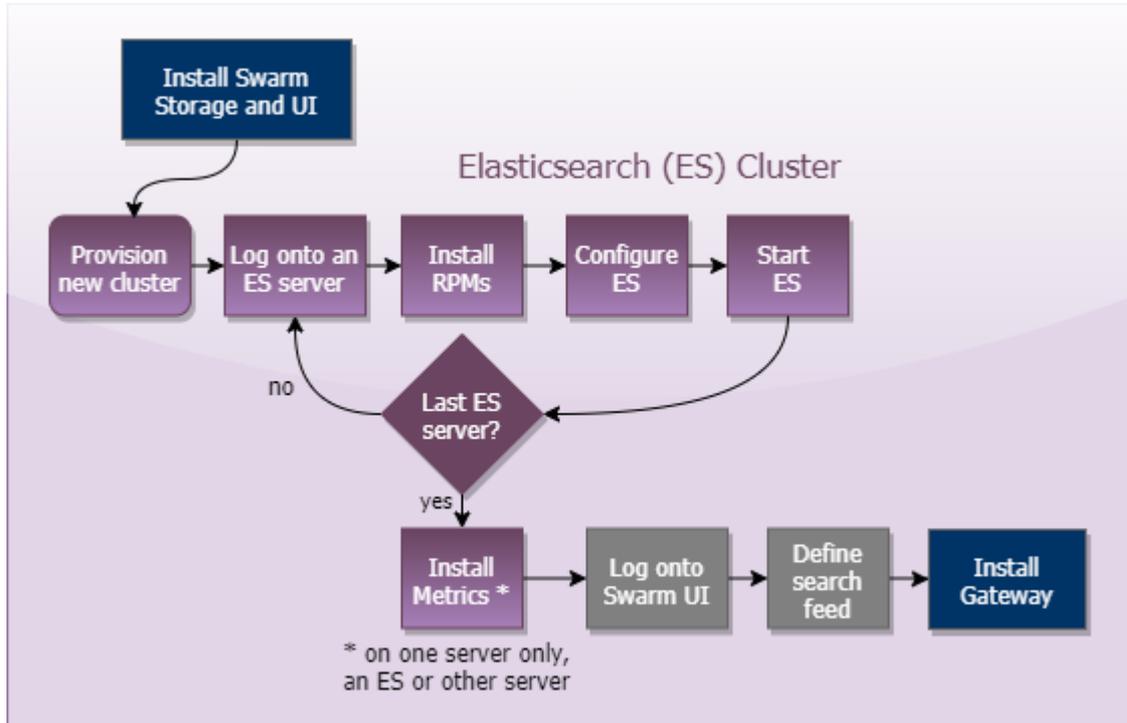
Elasticsearch 7 bundles Java so there is no need to install OpenJDK for it. Use the below command to remove that installed OpenJDK if already installed earlier for Elasticsearch 6.

Remove OpenJDK Package

```
yum remove java-1.8.0-openjdk
```

Installing Elasticsearch

This is an overview of the Elasticsearch (ES) installation process. For upgrading Elasticsearch, see [Upgrading Elasticsearch](#).



1. Prepare for installation.

- a. From the Swarm bundle download, get the latest Elasticsearch RPM and Swarm Search RPM, which installs plugins and support utilities.

```

elasticsearch-VERSION.rpm
caringo-elasticsearch-search-VERSION.noarch.rpm
    
```

- b. Install the Caringo RPM public key that is included with the distribution bundle by running the following command:

```

rpm --import RPM-GPG-KEY
rpm --import GPG-KEY-elasticsearch
    
```

2. On each ES server, install and configure the Elasticsearch components.

- a. Install the RPMs. Do not attempt to install with `rpm` - it does not install dependencies like `python3`.

```

yum install elasticsearch-VERSION.rpm
yum install caringo-elasticsearch-search-VERSION.noarch.rpm
    
```

- b. Complete configuration of Elasticsearch and its environment. See [Configuring Elasticsearch](#). The configuration script starts the Elasticsearch service and enables it to start automatically.

Single-node ES cluster

If you are implementing a single-node ES cluster, you need to set the number of replicas to zero to avoid yellow status from having too few nodes. See [Scaling Elasticsearch](#).

- c. Verify the `mlockall` setting is true. If it is not, contact DataCore Support.

```
curl -XGET "ES_HOST:9200/_nodes/process?pretty"
```

d. Proceed to the next server.

3. At this point, all ES servers should be installed and started. Use one of these methods to verify Elasticsearch is running (the status is yellow or green):

```
curl -XGET ES_HOST:9200/_cluster/health
systemctl status elasticsearch
```


Tip

When troubleshooting Elasticsearch issues, run the status command (`systemctl status elasticsearch`) and then look at the log entries:

```
/var/log/elasticsearch/CLUSTERNAME.log
```

4. In the Swarm UI, create a new search feed. (See [Managing Feeds.](#))


Best practice

Do not skip creating a search feed if you have an available Elasticsearch cluster with enough resources. A search feed simplifies enumerating your buckets/domains/clusters and making use of your valuable metadata.

Indexing is performed when the feed shows 0 "pending evaluation".


Tip

To set up the ability to restore search data on demand, see [Snapshot and Restore Search Data.](#)

Hardware Requirements for Elasticsearch

- [Hardware Best Practices](#)
- [RAM for Elasticsearch](#)
- [Disk Usage for Search](#)
- [Optimizing Disk I/O for ES](#)
- [Optimizing Disaster Recovery for ES](#)

An Elasticsearch cluster supports Swarm searches. The Swarm feeds mechanism (see [Understanding Feeds](#)) populates the metadata search servers running the Elasticsearch (ES) software.

See [Elasticsearch Implementation](#).

Info

Elasticsearch was previously used to store Historical Metrics but has moved to [Prometheus](#) starting with Swarm 14 . Gateway [Content Metering](#) stores *csmeter indices* in Elasticsearch, but this does not impact Elasticsearch hardware requirements as much as a Swarm Search Feed.

This software requires one or more servers running RHEL/CentOS 7 Linux. Although Elasticsearch runs on other Linux platforms, DataCore currently provides support and testing for these versions. The Elasticsearch version provided with the Swarm distribution is supported.

See the [Elasticsearch project website](#) for more about Elasticsearch.

Do not install on management node

Both the Content Gateway and the production Elasticsearch cluster need to be on separate machines from the management node (SCS). The management node installs with Service Proxy and a single-node ES, which are dedicated to the Swarm UI.

Hardware Best Practices

Following are overall best practices, with [hardware recommendations](#) from Elasticsearch:

- Provision the machines with CPUs with at least 4 cores and 64 GB memory. Between faster processors or more cores, choose more cores.
- Choose Solid-state drives (SSD), to boost performance. This is critical for S3, especially rapid small object writes, and for the listing of buckets with millions of objects.
- Perform the following if using hard disks which do not handle concurrent I/O as well as SSDs:
 - Select high-performance server disks.
 - Use RAID 0 with a writeback cache.
 - Set `index.merge.scheduler.max_thread_count` to 1, to prevent too many merges from running at once.

```
curl -X PUT <ES_SERVER>:9200/<SWARM_SEARCH_INDEX>/_settings \
-d '{ "index": { "merge.scheduler.max_thread_count": 1 } }'
```

- As with the storage cluster, choose similar, moderate configurations, for balanced resource usage.

RAM for Elasticsearch

RAM is key for Elasticsearch performance. Use these guidelines as a basis for capacity planning:

- 64 GB RAM per machine is optimal ([recommended by Elasticsearch](#)).
- Dedicate half of total RAM to the Java Virtual Machine (JVM) running Elasticsearch, but [do not exceed 31 GB](#), for best performance.
- Disable swapping of the Elasticsearch image. (For ES 2.3.3, allow in-memory caching of all shards on the server.)

Optimal performance can be achieved by adding adequate RAM in the ES servers to store all database shards in memory. Take steps to [disable or mitigate swapping](#). Memory page swapping on an ES server impacts Elasticsearch performance.

ⓘ Important

Watch for sustained increases in page swapping and disk I/O when monitoring the ES servers. This may mean additional RAM is needed on an ES server or additional servers need to be deployed to offset the load.

Disk Usage for Search

The storage on the Elasticsearch servers is used to persist the shards of the Swarm Search. Follow these guidelines for capacity planning for the Swarm Search indices.

- **Baseline metadata** to support listing: 150 GB per 200 million objects
- **Full metadata** to support ad-hoc searching: 300 GB per 200 million objects
- **Custom metadata**: allocate additional storage in proportion if indexing a large amount of custom metadata

These are unique *objects*, not *replicas*: how many Swarm replicas a Swarm object has is irrelevant to the ES servers. There is one metadata entry for the object no matter how many replicas of an object exist in the cluster.

ⓘ Tip

Do not confuse this with the RAM-based [Overlay Index](#) each *storage node* maintains, which depends on the total number of replicas in the cluster.

Optimizing Disk I/O for ES

Elasticsearch heavily utilizes disks, so higher throughput results in more stable nodes. Follow these [Elasticsearch guidelines](#) for optimizing disk I/O:

- **Use SSDs.** SSDs boost performance. With SSDs, verify the [OS I/O scheduler is configured correctly](#).
- **Use RAID 0.** Striped RAID increases disk I/O, at the expense of potential failure if a disk dies. Do not use mirrored or parity RAIDS, because replicas provide this functionality.
- **Do not use remote-mounted storage**, such as NFS or SMB/CIFS; the latency negatively impacts performance.
- **Avoid virtualized storage, such as a SAN or EBS** (Amazon Elastic Block Store). Even when SSD-backed it is often slower than local instance storage and it conflicts with the purpose of replicas and sharding.

Optimizing Disaster Recovery for ES

Elasticsearch clustering is designed to mitigate the impact of hardware and power failures, so long delays from refreshing the search data are not experienced. Determining how to invest and optimize hardware depends on how important metadata search and querying is to the organization and how long these features can be offline while Elasticsearch rebuilds data.

These are principles for making a configuration more disaster-proof:

- Do not use and rely on a single Elasticsearch server. This introduces vulnerabilities and potential disruption of search capabilities, and risks too little capacity to support all search needs.
- For power failure protection, deploy enough Elasticsearch servers to survive multiple server failures and distribute them across different power sources.
- Set up Elasticsearch with multiple nodes spread equally among the subclusters if the cluster is divided in to subclusters to match the power groups. This strategy improves survivability of a power group failure.

Scaling Elasticsearch

The hardware platform for the Elasticsearch servers can scale from one virtual machine to multiple physical machines. Scaling Elasticsearch involves both the number of *nodes* and the number of *shards* for each index.

- [Number of Nodes and Shards](#)
 - [Best practices](#)
- [Increasing the Shards](#)
- [Single-Node ES Clusters](#)
 - [Important](#)

Number of Nodes and Shards

Elasticsearch data is organized in to *indices*, which are made up of one or more *shards*. Each shard is an instance of a Lucene index, which is like a self-contained search engine that indexes and handles queries for one specific portion of Elasticsearch data. Elasticsearch distributes these shards across the cluster of nodes. Refer to the [Elasticsearch guidelines](#).

Scaling is critical: Having too few *nodes* slows indexing and searching, and too few *shards* can make them expand excessively large (over 50 GB), which impinges the cluster's ability to recover from failure.

i Best practices

- As possible, add more Elasticsearch nodes, for faster indexing and searching. Nodes require RAM and SSD resources, but are easy to configure and join to the cluster.
- Do not exceed 200M documents (unique objects) per ES node. Exceeding this limit negatively impacts performance.
- Boost the shard count if expecting multi-billion objects or a lot of custom metadata. Review [Elasticsearch shard sizing guidelines](#) and consult DataCore Support for a recommended number.

Max objects *	ES nodes	Shards	Notes
200M	1	5	Must set shard <i>replicas</i> to zero (see below) Do not use without a robust ES backup strategy
600M	3	5	This is the default shard count
1000M	5	10	
1600M	8	15	
2000M	10	20	

* *Very large object sizes may have the effect of many more objects.* An application creates an equal number of Swarm objects if an application backs up large VM snapshots to Swarm. A large object count in Swarm is created, needing a larger ES cluster, if the backup application is chunking and storing every 5MB of each snapshot as a separate Swarm object (which has separate metadata).

Increasing the Shards

The Swarm setting `search.numberOfShards` allows adjusting the number of shards on new search indices as the implementation is scaled. The setting has no effect on existing indices. (v12.0)

To adjust the shard number, update the configuration and start a new feed based on the new shard count.

1. Set `search.numberOfShards` to the new value in the `cluster.cfg`.
2. Reboot the cluster, or apply it to the running cluster using the command in the Support Tools bundle, as directed by Support:

```
swarmctl -a -C search.numberOfShards -V 20
```

3. Create a new search feed using the new shard count:

- *No listing downtime*: Create a new feed, make it **Primary** once it completes, and then delete the old one.
- *With listing downtime*: Delete the existing ES index and **Refresh** the feed.

Single-Node ES Clusters

The default shard protection (1 *replica*) causes a permanent yellow status when an Elasticsearch cluster has one node. The yellow status occurs because the primary and replica shards are recommended to be hosted on *separate* nodes. To deploy a single-node ES cluster, explicitly change the shard replicas to zero.



Important

Change the replicas to zero *after* installing the ES RPMs and *before* configuring feeds, metrics, or metering, and implement a [robust ES backup strategy](#).

Update the `elasticsearch.yml` files with the new hosts and change the shard replicas back to 1 if deploying additional nodes later.

Configure the indices to have zero replicas using the script provided if configuring an ES cluster with one ES data node:

```
/usr/share/caringo-elasticsearch-search/bin/configure_replicas.py -r 0 -e <ES-SERVER>
```

To view the complete options for changing the number of replicas on existing indices, use the help command:

```
/usr/share/caringo-elasticsearch-search/bin/configure_replicas.py --help
```

Upgrading Elasticsearch

This is the process for in-place upgrades of Elasticsearch (ES), using an existing Search feed and index data.

i Required

- This upgrade is for Elasticsearch 6.8.6 and higher with a Search feed created on Swarm 11.
- See [Migrating from Older Elasticsearch](#) for migrating from Elasticsearch 2.3.3 or 5.6.12.

Upgrading Elasticsearch by script

On *each node* in an Elasticsearch cluster, follow this process and run the files from the Swarm download bundle:

1. Query the running Elasticsearch cluster, before upgrading, for the list of nodes.

```
curl -i http://ELASTICSEARCH:9200/_cat/nodes
```

Note which node is starred. That is the Elasticsearch master node which is recommended to upgrade *last* to avoid problems electing a new one.

2. Backup the existing `elasticsearch.yml`, so a record of any customizations made exists.
3. Create a symbolic (soft) link: `symlink /var/lib/elasticsearch` if `path.data` is customized . Perform the upgrade manually, as described below if unable.
4. Start by installing the latest Swarm Search, which is the `caringo-elasticsearch-search` RPM when upgrading Elasticsearch 6.

```
yum install caringo-elasticsearch-search-VERSION.noarch.rpm
```

i Tip

The error: "ES_PATH_CONF must be set to the configuration path chown: cannot access '/etc/elasticsearch/elasticsearch.keystore': No such file or directory" displays if Elasticsearch 7 RPM is inadvertently installed. Install `caringo-elasticsearch-search-7.0.0` RPM to proceed.

5. Run the script that installs and configures the upgrade.

The script detects Elasticsearch 6 is installed and `discovery.zen.unicast.hosts` is configured, so it runs as with `--upgrade` instead of configuring a new cluster.

```
/usr/share/caringo-elasticsearch-search/bin/configure_elasticsearch_with_swarm_search.py
```

6. Compare the backup file to the newly created `elasticsearch.yml` and add back any customizations needed, such as `network.host` (which defaults to `_site_`, all interfaces).
7. Verify all nodes are accounted for, all shards are assigned, and the status is green.

```
curl -i 'http://ELASTICSEARCH:9200/_cat/health?v'
```

The script updates the configuration files and restarts the service if Elasticsearch 7 is already installed.

i Troubleshooting

Change permissions if the Elasticsearch service fails and `journalctl -u elasticsearch` shows access is denied (`BootstrapException/AccessDeniedException`):

```
chown elasticsearch /etc/elasticsearch
```

Important

Type **Ctrl-C** once, when the upgrade script is stuck in retrying status checks and proceed to the next node after the script finishes if the cluster loses master during the upgrade process and does not recover. Review `/etc/elasticsearch/elasticsearch.yml` and `tail/var/log/elasticsearch/<cluster-name>.log` for configuration errors.

The nodes elect the master and recover once all nodes have started on Elasticsearch 7.5.2. Health status goes yellow then eventually green. Re-enable shard allocation, otherwise `/_cat/health?v` stops at 50% with health status yellow.

Upgrading Elasticsearch manually

These are the steps the script automates if needing to upgrade manually:

1. It fixes `/etc/sysconfig/elasticsearch` to the correct ES6 version (the same as ES7).
2. It increases the `systemd` timeout in `/etc/systemd/system/elasticsearch.service.d/override.conf` (see github.com/elastic/elasticsearch/issues/60140)
3. A prompt to continue with the yum upgrade to 7.5.2 appears after refreshing the config files for Elasticsearch 6.
4. It disables shard allocation and does a POST synced-flush for safer rolling upgrades.

Important

Disabling shard allocation or sync-flush can fail to contact the node, but do not proceed to upgrade the next node until the cluster health is green again.

5. It uninstalls the Prometheus Exporter plugin if it exists.
6. It shells out to yum to install the Elasticsearch 7 RPM in the current directory or from artifacts.elastic.co, if unavailable.

! Internet access is expected for the upgrade. Verify the Elasticsearch RPM is in the current directory if access is unavailable.

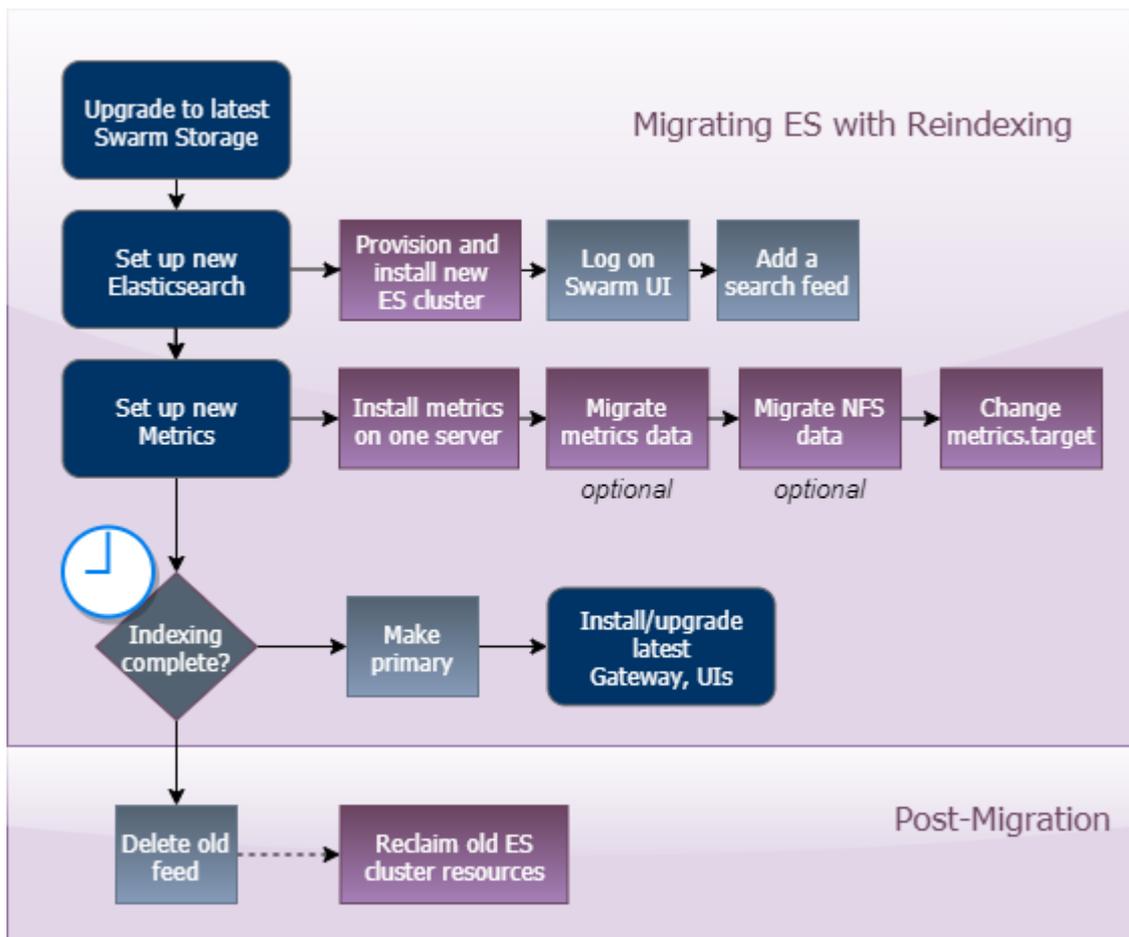
7. It updates `elasticsearch.yml` for version 7 compatibility, including `discovery.initial_master_nodes` instead of `discovery.zen.unicast.hosts`, and `jvm.options`.
8. It starts the upgraded Elasticsearch 7 and waits for it to be ready.
9. The cluster re-enables shard allocation and prompts to repeat these two steps on the next node if the cluster health is green or yellow.

Migrating from Older Elasticsearch

- [Pre-Upgrade Checklist](#)
- [Migrating to Elasticsearch 6](#)
 - [Important](#)

Newer versions of Elasticsearch include advances in speed, security, scalability, and hardware efficiency, and they support newer tool releases. All upgrades from Elasticsearch 2.3.3/5.6.12 to version 6.8.6 are legacy migrations requiring both new separate Elasticsearch cluster and a new Swarm Search Feed to reindex content into the new ES cluster and format. Upgrade in-place to Elasticsearch 7 as part of the Swarm 12 upgrade once on ES 6.8.6.

Migration process – Given the complexities of converting legacy ES data, the easiest path is to start fresh: provision a new ES cluster (machines or VMs meeting the [requirements](#)), install ES, Search, and Swarm Metrics on this cluster, and create a new search feed to this cluster. Swarm continues to support the existing primary feed to the legacy ES cluster while it builds the index data for the new feed. Searching remains available for users. Make the new feed primary, restart the Gateways, and the migration is complete once the new feed has completed indexing. This is an overview of the migration process to Elasticsearch 6:



- [Pre-Upgrade Checklist](#)
- [Migrating to Elasticsearch 6](#)
 - [Important](#)

Pre-Upgrade Checklist

Swarm Requirements	<ol style="list-style-type: none"> 1. Upgrade Swarm Storage – Complete the upgrade to the latest version of Swarm Storage. See How to Upgrade Swarm. 2. Case-sensitivity – Content Gateway still allows S3 to perform the case-sensitive operations it needs if enabling case-insensitive searching in SCSP (<code>search.caseInsensitive = 1</code>). 3. (optional) Enable atime – Enable Time of Last Access (<code>atime</code>) feature now so the new index populates the <code>accessed</code> field if implementing it, which requires a full reindexing. The feature does affect performance, so review the impact discussion here: Time of Last Access - atime.
New ES Cluster	<ol style="list-style-type: none"> 1. Provision a new set of ES servers (machines or VMs) on which to install the new version of Elasticsearch. Do not attempt to upgrade the legacy ES servers: it is challenging to clean up old data and config files. <ul style="list-style-type: none"> • Contact DataCore Support for assistance if provisioning a new ES cluster is not possible. 2. Verify every Elasticsearch node meets the hardware, network, and software requirements, including the latest RHEL /CentOS 7 and Java 8. <p>See Hardware Requirements for Elasticsearch.</p> <p>See Preparing the Search Cluster.</p>

Migrating to Elasticsearch 6

Follow these steps to migrate to a new Elasticsearch 6 cluster, from which allow upgrades to Swarm 12 and Elasticsearch 7 in-place, retaining the same Search feed and index data. An existing Elasticsearch 2.3.3 or 5.6.12 cluster cannot be upgraded; a new cluster and new Search Feed must be created.

Important

*Do not run different versions of Elasticsearch in the same ES cluster. Verify the new Elasticsearch configuration has a *different name* for the new cluster; otherwise, the new ES servers join the old ES cluster.*

1. Set up the new Elasticsearch.

- a. Obtain the Elasticsearch RPMs and Search from the downloaded Swarm bundle.
- b. On each ES server in the newly provisioned cluster install and configure the Elasticsearch components.

- i. Install the RPMs from the bundle.

```
yum install elasticsearch-VERSION.rpm
yum install caringo-elasticsearch-search-VERSION.noarch.rpm
```

- ii. Complete configuration of Elasticsearch and the environment. See [Configuring Elasticsearch](#). Set the number of replicas to zero to avoid warnings if using a single-node ES cluster. See [Scaling Elasticsearch](#).
- iii. The configuration script starts the Elasticsearch service and enables it to start automatically.
- iv. Verify the **mlockall** setting is true. Else, contact DataCore Support.

```
curl -XGET "ES_HOST:9200/_nodes/process?pretty"
```

- v. Proceed to the next server.

- c. All ES servers are installed and started at this point. Use Swarm UI or one of these methods to verify Elasticsearch is running (the status is yellow or green):

```
curl -XGET ES_HOST:9200/_cluster/health
systemctl status elasticsearch
```

Troubleshooting – Run the status command (`systemctl status elasticsearch`) and look at the log entries: `/var/log/elasticsearch/CLUSTERNAME.log`

2. Create a search feed for the new ES. Swarm allows creation of more than one Search feed supporting transition between Elasticsearch clusters.

- a. Create a new search feed pointing to the new Elasticsearch cluster in the Swarm UI. See [Managing Feeds](#).
- b. *Continue using the existing primary feed for queries* during the transition. The second feed is incomplete until it fully clears the backlog.

3. Set up Swarm Metrics.

- a. Install Swarm Metrics on *one* server in the new Elasticsearch cluster (or another machine running RHEL/CentOS 7). See [Installing Swarm Metrics](#).
- b. (*optional*) Swarm Metrics includes a script to migrate the historical metrics and content metering data. Proceed with the following steps if preserving the historical chart history is required (such as billing clients based on storage and bandwidth usage):

- i. Add a "whitelist" entry to the new ES server so it trusts the old ES server before running the script.

1. Edit the config file: `/etc/elasticsearch/elasticsearch.yml` on the destination ES node.
2. Add the whitelist line, using the old ES source node in place of the example:

```
reindex.remote.whitelist: old-indexer.example.com:9200
```

3. Restart Elasticsearch: `systemctl restart elasticsearch`

- ii. Run the data migration script, specifying the source and destination clusters:

```
/usr/share/caringo-elasticsearch-metrics/bin/reindex_metrics -s ES_OLD_SERVER -d
```

Troubleshooting options:

- By default, the script includes all metering data (client bandwidth and storage usage). To skip importing this data, add the flag `-c`.
 - To force reindexing of all imported data, add the flag `--force-all`.
- iii. Allow an hour or more for the script to complete if there are a large amount of metrics to convert (many nodes and several months of data).
 - iv. Run the script again, and repeat until it completes successfully if connection or other problems occur and the screen reports errors.
 - v. To see the past metrics, prime the curator by running it with the `-n` flag:

```
/usr/share/caringo-elasticsearch-metrics/bin/metrics_curator -n
```

- c. Change the `metrics.target` from the old ES target to the new ES target. This reconfiguration pushes the new schema to the new ES cluster.

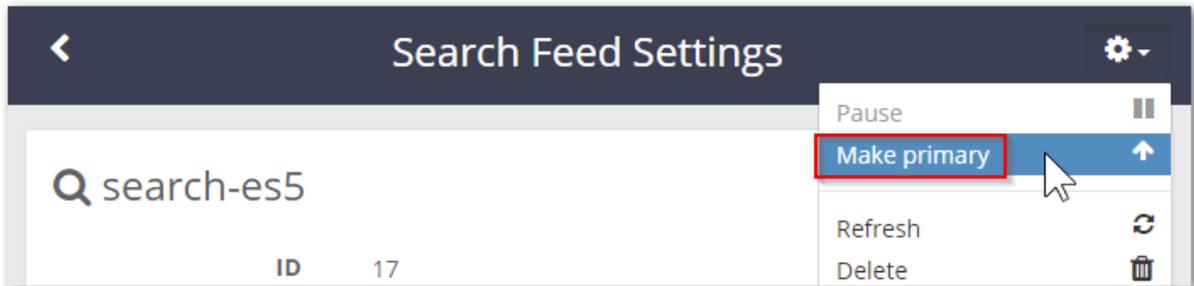
4. Complete new feed and make primary.

Important

Migration is not done until Swarm switches to using the new ES search feed. Because it takes days for a large cluster to reindex the metadata, set calendar reminders to monitor the progress.

- a. On the Swarm UI's **Reports > Feeds** page, watch for indexing to be done, which is when the feed shows 0 "pending evaluation".

- b. Set Swarm to use the second feed when it is caught up. In the feed's command (gear) menu, select **Make Primary**.



5. **Install Gateway 7.0, Swarm UI, and Content UI** on each Gateway server.

- For new Gateway servers, see [Gateway Installation](#).
- For Gateway 6.x servers, follow [Upgrading Gateway](#).
- For Gateway 5.x servers, follow [Upgrading from Gateway 5.x](#).

6. **Complete post-migration.**

- a. Delete the original feed when verifying it is working as the new primary feed target. Having two feeds is for *temporary use* because every feed incurs cluster activity, even when paused.
- b. As appropriate, decommission the old ES cluster to reclaim those resources.

Upgrade in place to Swarm 12, which includes Elasticsearch 7 when completing this migration and are running Swarm 11.3. See [How to Upgrade Swarm](#).

Content Gateway Implementation

See also these sections:

- [Content Gateway Release Notes](#)
- [Swarm Content Gateway](#) (usage guide)
- [Content Application Development](#)
- [S3 Protocol Interface](#)

System Requirements

This section covers the hardware requirements and software installation and configuration of the Gateway platform components, for those in the following roles who need to deploy and manage the Gateway:

- Storage system administrators
- Network administrators
- Technical architects

The administrators are normally responsible for allocating storage, managing capacity, monitoring storage system health, replacing malfunctioning hardware, and adding additional capacity when needed. Network administrators are responsible for TCP/IP switching, routing, load balancing, and firewall setup. Herein, these different roles are referred to as *administrators*.



Note

This section assumes familiarity with the deployment and management processes for [Swarm Administration](#) and have knowledge of TCP/IP networking, basic x86 hardware setup, and intermediate Linux system administration.

- [Gateway Logging](#)
- [Gateway Installation](#)
- [Gateway Administrative Domain](#)
- [Gateway Verification](#)
- [Gateway Configuration](#)
- [Gateway Requirements](#)
- [Configuring Swarm Storage for Gateway](#)

Gateway Logging

The `logging.yaml` file is a standard log4j2 configuration file installed with Gateway.

Upgrading from Gateway 5.x

The configuration file for logging changed from `logging.cfg` (log4j) to `logging.yaml` (log4j2) to support Elasticsearch 5+. Add customizations to the YAML file provided. (v6.0)

Back up the existing config – When upgrading, save copies of any prior configuration files. Use a copy command:

```
cp /etc/caringo/cloudgateway/logging.yaml{,.bak}
```

1. On the Gateway:

- `/etc/caringo/cloudgateway/logging.yaml`
- `/etc/logrotate.d/caringo-content-gateway-*`

2. On the syslog server, such as CSN:

- `/etc/rsyslog.conf`
- [Upgrading from Gateway 5.x](#)
- [System Logging](#)
- [Audit Logging](#)
- [Syslog Setup](#)
 - [Troubleshooting](#)

System Logging

The Gateway's system logs record operational details about the execution of the Gateway; they assist administrators and DataCore Support to monitor and troubleshoot issues.

Log levels – Editing the `logLevel` property in the `logging.yaml` file changes the logging level in `/var/log/caringo/cloudgateway_server.log`. There is no need to restart for the new level to take effect because the file is checked for changes every few seconds. The `logLevel` property is located at the top of the `logging.yaml` file for easy access as of Gateway 7.1. (v7.1)

These are the levels available:

- trace
- debug
- info (default)
- warn
- error

Rolling files – This extract from the `logging.yaml` file shows logging directly to the Gateway server's file system, rolling log files when they reach 100 MB in size, and keeping 10 generations of rolled log files:

```
RollingFile:
  # A file appender
  - name: file
    fileName: ${logpath}/cloudgateway_server.log
    filePattern: ${logpath}/cloudgateway_server.log.%i
    # filePattern: "${logpath}/cloudgateway_server.%i.log.gz"
    PatternLayout:
      pattern: ${logPattern}
    Policies:
      SizeBasedTriggeringPolicy:
        size: 100 MB
    DefaultRollOverStrategy:
      max: 10
  ...
```

Timestamps in filenames – Adding the field `%d{ISO8601}` in the conversion pattern tells Gateway to provide its own timestamp value:

```
pattern="%d{ISO8601}{GMT} %p [%X{requestId}] %msg%n"
```

Audit Logging

The audit logging data feed records client requests to the storage system in a well-defined format that is suitable for automatic processing by billing and compliance applications. The definition of this format is documented in [Gateway Audit Logging](#).

The audit logging is configured within the `logging.yaml` file along with the system logging configuration. The audit logging looks for `name: audit` to separate messages from system messages, so do not change it.

This extract from the `logging.yaml` file shows audit logging directly to the Gateway server's file system, rolling log files when they reach 100 MB in size, and keeping 10 generations of rolled log files:

```
RollingFile:
  ...
  # Audit log appender - DO NOT MODIFY!!!
  - name: audit
    fileName: ${logpath}/cloudgateway_audit.log
    filePattern: ${logpath}/cloudgateway_audit.log.%i
    # filePattern: ${logpath}/cloudgateway_audit.%i.log.gz
    PatternLayout:
      pattern: "%d{ISO8601}{GMT} %p [%X{requestId}] %msg%n"
    Policies:
      SizeBasedTriggeringPolicy:
        size: 100 MB
    DefaultRollOverStrategy:
      max: 10
```

Filtering event types – Audit messages are logged at INFO level, and each message event type can be filtered separately; setting the filter threshold to WARN, ERROR, or FATAL disables output of the message event type. Change the filtering threshold to INFO to enable and FATAL to disable the type for direct control of the message event types.

See [Gateway Audit Logging](#) for information about message event types and the format of the audit log messages.

Syslog Setup

Send Gateway's audit and server log messages to a file local to each Gateway server, or send them to a centralized syslog server, such as the CSN. The following changes need to be made To send logs to a syslog.

Enable syslog. Make the following changes in the `logging.yaml`.

1. Set the "host" in both places (under `audit_syslog` and `server_syslog`) to the syslog host.

2. Uncomment the lines “`ref: audit_syslog`” and “`ref: server_syslog`”.

Configure syslog. Configure the syslog server with the location to write these log files.

1. Navigate to the CSN's `/etc/rsyslog.conf` and add these two lines in the `local*` section at the bottom of the file:

```
local4.* /var/log/caringo/cloudgateway_server.log
local5.* /var/log/caringo/cloudgateway_audit.log
```

The result resembles:

```
# ### end of the forwarding rule ###
$ModLoad imudp.so
$SystemLogRateLimitInterval 0
local3.* /var/log/caringo/scspproxy.log
local6.* /var/log/caringo/castor.log
local0.* /var/log/caringo/csn.log
local4.* /var/log/caringo/cloudgateway_server.log
local5.* /var/log/caringo/cloudgateway_audit.log
$UDPServerRun 514
```

2. Save the file and restart rsyslog.

Enable rotation. Create these two rotation files and add them to `/etc/logrotate.d/`. See the [Apache documentation](#) for logging details.

Caringo-content-gateway-server

```
# Content Gateway Server logrotate.d file
#
/var/log/caringo/cloudgateway_server.log {
    weekly
    rotate 30
    size 512M
    compress
    missingok
    copytruncate
}
```

Caringo-content-gateway-audit

```
# Content Gateway Audit logrotate.d file
#
/var/log/caringo/cloudgateway_audit.log {
    weekly
    rotate 30
    size 512M
    compress
    missingok
    copytruncate
}
```

Verify the syslog entries. Check the places where it defines the syslog in `logging.yaml`.

1. An IP address such as 192.168.203 needs to include the `bond0` interface of the CSN at that site; the bond interface should end in `.5`:

```
Syslog:
  - name: gateway
    host: 192.168.203.5
  ...
  - name: gateway_audit
    host: 192.168.203.5
```

2. Nothing needs to be restarted or the logging changes to take effect.

3. Check `/var/log/caringo/` for entries for the Gateway's logs on a CSN; entries start to display.

Back up the logging config. Save copies of the edited and created files. Use a copy command:

```
cp /etc/caringo/cloudgateway/logging.yaml{,.bak}
```

1. On the Gateway:

- `/etc/caringo/cloudgateway/logging.yaml`
- `/etc/logrotate.d/caringo-content-gateway-*`

2. On the syslog server, such as CSN:

- `/etc/rsyslog.conf`



Troubleshooting

Diagnosing configuration problems for logging can be challenging. Check `/var/log/messages` for errors from "startgateway" if the expected log files are not present.

Gateway Installation

The Content Gateway is the access point and gatekeeper for the back-end storage cluster. It provides value-added services for user applications and storage administrators.

SCSP Proxy

See [Migrating from SCSP Proxy](#) implementing Gateway to replace SCSP Proxy.

Important

Plan to have one Gateway dedicated to run as [Service Proxy](#) for your cluster administration (using Swarm UI and Management API) in production, and have a pool of additional Gateway appliances to handle all content management at scale. Enable *both* cluster administration and content management on a single Gateway instance if the cluster is for testing or light usage.

Upgrading

See the **Upgrading** section in [Release Notes](#) for the version being installed for information about upgrading.

1. Apply all current operating system patches *before* installing the Gateway. The Installer preserves and existing versions of `pip` and `requests` detected.
2. Install Java: Gateway requires a Java 8 SDK. The SDK must be Oracle's Java or OpenJDK. *Use the same JVM and version on all Gateway appliances.*

- a. Install the Java language on the server:

```
yum install java-1.8.0-openjdk
```

- b. After installing, verify the correct Java version is active:

```
java -version
```

- c. If you need to change the active Java version, run the following command:

```
alternatives --config java
```

3. Download the Swarm bundle from the [Downloads section](#) on the [DataCore Support Portal](#) to get the Gateway distribution, and unzip it.
4. Locate the RPM for the Gateway software. If you have not previously added the Caringo RPM public key that is included with the distribution bundle to your system, run the following command:

```
rpm --import RPM-GPG-KEY
```

5. Run the following command to install the Gateway package, substituting the exact version number for the RPM in the distribution file for the {version} string:

```
yum install caringo-gateway-{version}.rpm
```

6. Navigate to the `examples` directory for configuration file examples to study and clone:

```
/etc/caringo/cloudgateway/examples
```

7. Complete authentication for Gateway. Note: plain-text passwords in both Gateway configuration and IDSYS are replaced by encrypted versions during Gateway's startup. (v7.1)

- a. Complete the [IDSYS document](#) for user authentication:

```
/etc/caringo/cloudgateway/idsys.json
```

- b. Complete the [Policy document](#) for access control:

```
/etc/caringo/cloudgateway/policy.json
```

8. Record the location for the server logs:

```
/var/log/caringo/cloudgateway_server.log
```

9. Verify NTP time synchronization is being used on the Gateway server to guarantee proper storage transaction handling and that the audit log timestamps match across servers. NTP is critical for the operation of Swarm and should be used on all hosts that interact with Swarm.
- [Migrating from SCSP Proxy](#)

Migrating from SCSP Proxy

- [Planning an SCSP Proxy migration](#)
- [Adding Authentication for Gateway](#)

Planning an SCSP Proxy migration

Consider which of these steps apply to the implementation if installing Gateway to replace SCSP Proxy:

1. Update applications to remove any `Expect: ContentMD5` headers, which are unsupported by Gateway and unneeded if `scsp.autoContentMD5Computation` is enabled (see [Content-MD5 Checksums](#)).
2. (optional) Create domains, and update applications to write to them. (See [Content Management API](#).)
 - Keep using `enforceTenancy=false` in Swarm Storage. (See [How enforceTenancy Works](#).)
 - Update applications to start adding `Content-type: application/castorcontext` when creating (via POST) a domain or bucket. (See [Managing Domains](#).)
3. (optional) Plan for a production-sized Elasticsearch cluster (See [Elasticsearch Implementation](#)) and add a Swarm Search Feed if the number of objects in the cluster is not too large to index.



Benefits of Search

Use Content UI and S3 (named objects only) with a search index, and Gateway metering can be enabled to track the content users' space and bandwidth usage, with optional quotas. (See [Swarm Content UI](#).)

Adding Authentication for Gateway

No configuration in `idsys` or `policy` is required when implementing Gateway to replace SCSP Proxy. Gateway's root `policy.json` provides full anonymous access, and the `idsys.json` is empty (no users) by default.

Enable PAM/LDAP users by configuring the root `idsys.json` and edit the root `policy.json` to permission `GetObject`, `PutObject`, etc. as needed if needing to grant specific read/write access to untenanted objects.

See [Content Gateway Authentication](#).

Gateway Administrative Domain

Content Gateway uses one storage domain within the storage cluster to persist meta information about all tenants and storage domains. Although there is no difference between storage domains to the storage cluster, Content Gateway uses these two distinctions for domains: *administrative domain*, *tenant storage domain*.

- **administrative domain** refers to the domain used by Gateway to store meta information used in the management of tenants and all other storage domains, including itself, and should only be accessible to cluster administrators. It is not recommended to use the administrative domain to store general-purpose content. Do not interfere with the objects managed by the Gateway.
- **tenant storage domain (or storage domain)** refers to the domains that store content that is accessible to normal users and applications. All content within a tenant storage domain is potentially accessible to the users of that domain and there is no special Gateway content within it.

The requirements for the name of the administrative domain:

- globally unique for a set of tenant storage domains
- defined in the `gateway.cfg` file
- created prior to using tenant storage domains
- same for all Gateway servers servicing a set of tenant storage domains



Important

The content within the administrative domain must be protected from access by users other than the cluster administrators. Thus, when this domain is created, an owner must be set and, optionally, an appropriate domain Policy should be defined for it.

Gateway includes a command to properly create a locked-down domain to facilitate the setup of the administrative domain. Edit the `gateway.cfg` file's `adminDomain` parameter to use the command. Define the name for the administrative domain and run the initialization script:

```
/opt/caringo/cloudgateway/bin/initgateway
```



Caution

Run once only. This command should be *run one time* when installing the first Gateway server; it should not run when installing subsequent servers.

Run locally only. Do not run the command in a remote cluster which replicates the administrative domain using a Feed.

A domain named by the `adminDomain` parameter is created in the storage cluster with the owner set to the value `admin@`. Without additional action on the part of the cluster administrator, this domain is locked for all access and requires the use of an administrative override to log in to the domain.

See [Restricting Domain Access](#) for more about access control and administrative override.

Cluster administrators use the Policy and IDSYS documents for the domain and change the ownership by modifying the `X-Owner-Meta` metadata value if access of the administrative domain needs to be open.



Caution

Verify access to the administrative domain is locked or unlocked. Content stored within the administrative domain controls access, policies, and management data for all tenants and storage domains.

The name of the administrative domain must be unique for a set of tenant storage domains and must not be created more than once whether using an SCSP operation or by using the `initgateway` script. Once an administrative domain or a tenant storage domain has been created, the only proper way to instantiate the domain in another cluster is by using remote replication in Swarm.

See [Replicating Domains to Other Clusters](#).

Gateway Verification

After configuring Content Gateway, determine whether it is working correctly with Swarm and Elasticsearch by performing a functional verification. Create a domain within Gateway, then a bucket under that domain, then an object in that bucket. Perform search query and read against the object. Create tokens for S3 client token authentication, and assign a Gateway tenant context to a domain.

- [Authentication](#)
 - [Note](#)
- [Create Domain, Bucket, and Object](#)
 - [Note](#)
- [Create Tokens for S3 Clients](#)
 - [Note](#)
- [Assign Tenancy to Domains under Gateway](#)

Authentication

The default install has an "anonymous can do anything" [policy.json](#) and an empty [idsys.json](#) (therefore, no users). Before proceeding with verification, set up authentication.

Note the following assumptions and requirements:

- The authentication store of choice as referenced in Content Gateway's root "[idsys.json](#)" is configured correctly and ready to handle authentication requests (LDAP, Active Directory, or local Linux PAM – pluggable authentication modules).
- The user designated as the top level "root" user for Content Gateway is named "admin" with password of "password".
- Gateway's root "[policy.json](#)" is properly configured to allow this user full access rights for operations to the entire cluster.



Note

These configuration files are found on the Gateway machine(s) under the directory `"/etc/caringo/cloudgateway"`.

See [Content Gateway Authentication](#).

Create Domain, Bucket, and Object

A domain and bucket in the domain are created then an object is instantiated in the bucket.



Note

Hostname / IP address information and, alternatively, service port need to be changed depending on how the routing to the Gateway machine is configured.

1. Domain creation of domain 'demodomain.caringodemo.int' in the cluster 'caringodemo.int' (Gateway target GATEWAY:PORT):

```
curl -v -u "admin:password" -X POST -d "" "http://GATEWAY:PORT/?domain=demodomain.caringodemo.int"
```

2. Bucket creation of 'bucket1' in the new domain 'demodomain.caringodemo.int':

```
curl -v -u "admin:password" -X POST -d "" "http://GATEWAY:PORT/bucket1?domain=demodomain.ca
```

3. Placing a file/stream/object 'install.log' in the bucket 'bucket1' within the domain 'demodomain.caringodemo.int':

```
curl -v -u "admin:password" -X POST --data-binary @install.log "http://GATEWAY:PORT/bucket1,
```

4. Listing the indexed bucket(s) and contents of bucket 'bucket1' in the domain 'demodomain.caringodemo.int':

```
curl -v -u "admin:password" 'http://GATEWAY:PORT/?format=json&domain=demodomain.caringodemo
```

```
curl -v -u "admin:password" 'http://GATEWAY:PORT/bucket1?format=json&domain=demodomain.car:
```

5. Retrieving the stream 'install.log' from bucket 'bucket1' in domain 'demodomain.caringodemo.int':

```
curl -v -u "admin:password" 'http://GATEWAY:PORT/bucket1/install.log?domain=demodomain.cari
> install.log
```

Create Tokens for S3 Clients

Proceed to token creation now that basic operations using Swarm SCSP are used to verify functionality. This allows for the creation of token /secret pairs for a given domain, which can then be assigned to S3 clients to allow them access via Content Gateway's S3 protocol.

These commands assume a user "myuser" is creating the necessary tokens to set up an S3 client for access.



Note

SCSP commands must be used to create tokens, so "GATEWAY-SCSP-ADDRESS:PORT" represents the Gateway and listening service port where the SCSP protocol interaction takes place.

1. Create a token for S3 token auth that expires Jan 1, 2020 at 00:00:01 hours (POSIX time converter at onlineconversion.com/unix_time.htm):

```
curl -v -X POST -H 'X-User-Secret-Key-Meta: MySecretKey' -H 'X-User-Token-Expires-Meta: 157'
```

2. List tokens for domain demodomain.caringodemo.int:

```
curl -v -X GET -u myuser@ 'http://GATEWAY-SCSP-ADDRESS:PORT/.TOKEN/?format=json&domain=demo
```

3. List the header (including secret key) info for a given token:

```
curl -I -u myuser@ 'http://GATEWAY-SCSP-ADDRESS:PORT/.TOKEN/VALID-TOKEN-NAME-HEX?domain=demo
```

The key part of the output for this is: ... X-User-Secret-Key-Meta: MySecretKey ...

An S3 client should be configured to use the hexadecimal token along with the assigned secret to authenticate using Gateway's S3 protocol service.

Assign Tenancy to Domains under Gateway

To use Gateway's tenant functionality, add the tenant context to any domain that is created within such a deployment. To assign tenancy at domain creation time, issue a domain creation request using the following parameters:

```
curl -X POST -u "admin:password" -d "" "http://GATEWAY:PORT/?domain=demodomain.caringodemo.int&c
```

The key parameter is `-H "x-tenant-meta-name: customerdemo"` – this is the header assignment designating the domain Created is assigned to the 'customerdemo' tenant context.

Gateway Configuration

- [Password security](#)
- [Configuring the Content Gateway](#)
 - [Minimum Configuration](#)
 - [Configuration Sections of gateway.cfg](#)
 - [Note](#)
 - [Tip](#)
- [Setting Ports for Docker or Proxies](#)
- [Enabling the Service Proxy](#)
 - [Important](#)

These configuration files reside on the system after installing the Content Gateway service:

```
/etc/caringo/cloudgateway/gateway.cfg
/etc/caringo/cloudgateway/logging.yaml
```

Logging – After completing the Gateway configuration, see [Gateway Logging](#). As of Gateway 6.0, the configuration file for logging changed from `logging.cfg` to `logging.yaml` to support newer versions of Elasticsearch, add customizations to the YAML file. See the [Apache documentation for logging](#).



Password security

Plain-text passwords in both Gateway Configuration and [IDSYS](#) are replaced by encrypted versions on startup. Enter new passwords and restart Gateway when management passwords need to be changed, which replaces those strings with encrypted versions as part of startup. (v7.1)

Configuring the Content Gateway

Minimum Configuration

While cluster administrators must understand the details of configuring Content Gateway, this section summarizes the minimum steps required to configure and run Gateway. To deploy Gateway in to production, additional customization is needed.

1. Check either that `IPTABLES` are off or that inbound access for the front-end protocols is allowed. These commands turn off and disable the firewall daemon.

```
systemctl disable firewalld
systemctl stop firewalld
```

2. Edit the `/etc/caringo/cloudgateway/gateway.cfg` file:
 - a. Set `adminDomain` to the name of an administrative domain that is created.
 - b. Set `hosts` for the storage cluster nodes. Including 3 to 5 nodes is sufficient for most deployments.
 - c. Set `indexerHosts` to the Elasticsearch servers (required for S3 and Content Metering).
 - d. Enable at least one of the front-end protocols: **SCSP** or **S3**.
Alternatively, for Service Proxy use (to host the Swarm UI), set *both* to disabled and complete the `[cluster_admin]` section.

3. Create the administrative domain by running the following on the first Gateway server:

```
/opt/caringo/cloudgateway/bin/initgateway
```

Password security – This one-time step initializes password encryption for the Gateway configuration and IDSYS files. If upgrading from a version prior to 7.1, this initialization must be run again on one Gateway server to enable the feature. (v7.1)

See [Gateway Administrative Domain](#).

4. Start the Gateway service:

```
systemctl start cloudgateway
```

5. Enable automatic startup of the Gateway service.

```
systemctl enable cloudgateway
```

Production deployments require customizations of the configuration parameters, below.

Configuration Sections of `gateway.cfg`

The `gateway.cfg` file controls the core operations of the Content Gateway. It is a plain text, INI-formatted file read when the Gateway is first started. The parameters within the file are organized in to the following sections, and colored rows are generally essential entries.

- [\[gateway\]](#)
- [\[storage_cluster\]](#)
- [\[scsp\]](#)
- [\[s3\]](#)
- [\[metering\]](#)
- [\[caching\]](#)
- [\[quota\]](#)
- [\[dynamic_features\]](#)

[gateway]

This section configures client communications:

adminDomain	gatewayAdminDomain	<p>Required. The administrative domain where meta information about tenants and storage domains is kept.</p> <p><i>Important:</i> This parameter must be set to the same value for all Gateway servers.</p> <p>This is <i>not recommended</i> to match the Swarm default domain (cluster.name).</p>
threads	200	<p>The number of threads allocated to handling client requests. Set for 100 times number of CPU cores. Minimum is 200.</p> <p>For CPUs with hyperthreading enabled, this calculation is based on the number of virtual cores, not physical.</p>
tokenTTLHours	24	<p>The default number of hours an authentication token is valid if no time is defined when it is created.</p>
multipartSpoolDir	<code>/var/spool /cloudgateway</code>	<p>The location of the spool directory for HTTP multipart MIME upload temporary space.</p> <p><i>Note:</i> Uploads through the Content UI use SCSP multipart uploads rather than multipart MIME uploads. (Gateway v6.2)</p>
multipartUsageAllowed	50	<p>The percentage of the file system that can be used for multipart MIME upload temporary space.</p>
recursiveDeleteMaxThreads	50	<p>The maximum number of parallel delete operations to dispatch when processing recursive delete requests.</p>
sanitizeErrors	false	<p>Set to true to hide identity management configuration details from authentication errors.</p>
cookieDomains		<p>One or more base domains for the <code>Set-Cookie</code> response header to scope (instead of the FQDN from the request) if an authentication token is created within a child domain of one of these base domains. This can be useful when using the Content UI to access multiple storage domains that share a common base domain when wanting to use the same authentication token across domains. (v5.2.2)</p> <p>Example:</p> <p><code>cookieDomains = cloud.example.com cloud.example.net</code></p>

[storage_cluster]

This section configures the back-end storage cluster:

locatorType	"static"	Zeroconf is not supported.
--------------------	----------	--

hosts	server1 server2 server3	Space-delimited list of IP addresses or host names of the storage cluster nodes.
port	80	Integer socket port number for SCSP on the storage nodes.
clusterName		The name of the storage cluster.
indexerHosts	indexer1 indexer2 indexer3	Space-delimited list of the Elasticsearch metadata index servers used by the storage cluster. <i>Must be from the same ES cluster: do not mix old and new clusters.</i> Required for the S3 protocol and for Content Metering . <i>Important:</i> If the <i>Primary Swarm Search</i> feed changes, update this setting and restart the Gateway servers. <i>indexerHosts</i> must match Swarm's default Search Feed because Gateway receives the index alias name from the default. <i>indexerHosts=ES5</i> does not work when set if the default feed is still <i>ES2</i> .
indexerPort	9200	The socket port on which the Elasticsearch servers listen.
managementPort managementUser managementPassword	91	Provide these credentials for the storage cluster to enable Gateway version and component information to be included in the cluster health report that provides proactive support from DataCore. (v6.0) Required when using [cluster_admin].
clientBindAddress	0.0.0.0	Set to the IP address of the network interface connected to the storage cluster subnet when using a multi-homed Gateway. The value must be defined as a non-default value when using a multi-homed Gateway server such as one connected to a front-end client network and a back-end storage network.
maxConnectionsPerRoute	100	The maximum number of open connections to a specific storage node.
maxConnections	250	The maximum number of open connections to allow. This includes both active and idle connections.
connectTimeout	60	The time in seconds allowed to connect to a node.
socketTimeout	120	The time in seconds allowed for an active connection to deliver data.
idleTimeout	120	The time in seconds an idle socket is allowed to remain in the connection pool.
indexerSocketTimeout	120	The time in seconds an indexer socket is allowed to remain in the connection pool. This affects the ability to list larger buckets. (v7.1) <i>Important:</i> Increase the load balancer (such as HAProxy) "timeout server" and S3 client timeouts as needed to match this.
continueWaitTimeout	30	The time in seconds to wait for client response after a 100 continue reply.

dataProtection	"immediate"	<p>Controls whether synchronous (<i>immediate</i>, via replicate on write) or asynchronous (<i>delayed</i>) data protection is requested when writing to the storage cluster.</p> <p>Values:</p> <ul style="list-style-type: none"> "immediate" (for replicate on write) – requires storage cluster setting of <code>scsp.replicateOnWrite=true</code> "delayed" (disables replicate on write) – requires storage cluster setting of <code>scsp.replicateOnWrite=false</code> <p>See Configuring ROW Replicate On Write.</p>
blockUndeletableWrites	true	<p>When enabled, the Gateway rejects any SCSP write (PUT, POST, COPY, APPEND) that includes a <code>deletable=no/false</code> lifepoint. This restriction applies to both named and unnamed (alias and immutable) objects. The request is refused with a 400 error message, "Unable to write undeletable object".</p>

[scsp]

This section configures the front-end SCSP protocol. This protocol must be enabled for any Gateway that services Content UI requests.

enabled	true	Activates this protocol: Values are: "true", "false".
bindAddress	0.0.0.0	The IP address of the network interface to which the listening socket binds. Defaults to all interfaces.
bindPort	80	Integer socket port number for protocol. <i>Important:</i> Must be unique from S3 port if both are enabled.
externalHTTPPort externalHTTPSPort	80 443	Optional, one or both. Allows Gateway to be used either behind a proxy or within a Docker environment, taking effect when X-Forwarded-Proto is found on the request. (Gateway uses X-Forwarded-Proto to determine which port to use.) (v5.4)
allowSwarmAdminIP	undefined	<p>Allows the use of internal Swarm requests for content replication to pass through the Gateway. This is useful if using replication feeds between clusters that use Gateway as the front-end.</p> <p>Values are "all", full IP addresses, IP address prefixes, a list of IPs/prefixes, or CIDR format such as 172.30.15.0/24.</p> <p>When undefined, no clients are allowed to send Swarm admin requests through the Gateway.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>This provides an additional layer of security. Any admin requests sent to Swarm require Swarm admin user and password correctly supplied.</p> </div>

[s3]

This section configures the front-end S3 protocol, which is optional.

enabled	false	The protocol must be explicitly enabled. Values are: "true", "false".
----------------	-------	---

bindAddress	0.0.0.0	The IP address of the network interface to which the listening socket binds. Defaults to all interfaces.
bindPort	80	Integer socket port number for protocol. <i>Important:</i> Must be unique from SCSP port if both are enabled.
externalHTTPPort externalHTTPSPort	80 443	Optional, one or both. Allows Gateway to be used either behind a proxy or within a Docker environment, taking effect when X-Forwarded-Proto is found on the request. (Gateway uses X-Forwarded-Proto to determine which port to use.) (v5.4)
enhancedListingConsistency	true	Improves compatibility with S3 clients and software libraries that expect consistent listings (despite the documented nature of listings to be eventually consistent). Can be disabled to boost write throughput (especially for small objects), if listing consistency is not critical. (v5.2.1) Exceptions to synchronous indexing: <ul style="list-style-type: none"> • Deletes of manifests for canceled multipart uploads are done asynchronously. • On a delete, when there is not enough space on the local node to write a delete marker for a named object, Swarm writes to another node and indexes asynchronously. • On a rename, Swarm indexes the new name synchronously, but the old name is deleted asynchronously. • On a parallel write complete, the init stream is deleted asynchronously.

[metering]

This section configures usage metering, which is optional. See [Content Metering](#).



Tip

Do not confuse Gateway *metering*, which manages quotas and graphs storage and bandwidth usage in Content UI, with [Swarm metrics](#), which provide graphs of cluster metrics in the Storage UI.

The Elasticsearch indices are distinguished by prefix:

- **csmeter-*** – Gateway metering
- **metrics-*** – Swarm historical metrics

enabled	false	The feature must be explicitly enabled.
flushIntervalSeconds	300 (5 minutes)	How frequently to send usage reports to Elasticsearch. Minimum is 10 seconds. The default value is optimized for the resolution of the queries.
retentionDays	100 (days)	How long to retain usage records. Minimum is 2 days. Allow for additional storage space if significantly increasing the retention period.
storageSampleIntervalSeconds	3600 (1 hour)	How frequently to sample the disk usage. Minimum is 900 (15 minutes). Larger values reduce the query workload on Elasticsearch.

[caching]

This section configures cache expiration. Times are in seconds. To disable, set to 0.

authRefresh	300	Time before authorization is revalidated with a request to the identity management system.
tokenRefresh	300	Time before an authentication token is revalidated with a request to the administration domain.
idsysRefresh	300	Time an IDSYS document is cached in memory.
policyRefresh	300	Time a tenant, domain, or bucket Policy document is cached in memory.
xformRefresh	300	Time an XFORM document is cached in memory.
metadataRefresh	300	Time that metadata for a tenant, domain, or bucket is cached in memory. This includes the owner for a tenant/domain/bucket and whether a bucket exists.
domainExistenceRefresh	300	Time that the knowledge of a domain's existence or nonexistence is cached.

[quota]

This section configures storage and network usage quotas. See [Setting Quotas](#).

The Gateway regularly refreshes the cache of quota information via an Elasticsearch query against usage metrics when enabled; it changes the quota state and performs the action specified by policy if any quota limit is reached.

enabled	false	The feature must be explicitly enabled.
minRefreshDeadline	60	The global limits on the speed of quota data refreshing. To increase the precision of the usage data, lower these values. To reduce the load on Elasticsearch, increase these values. To optimize the load on Elasticsearch, Gateway refreshes with a dynamic algorithm: slower when metrics are still far from the limit and faster when the limit approaches, slower when approaching a limit and faster as the overage nears an end. The minimum and maximum deadlines refer to the caps to apply to this refresh rate (no faster and no slower than these values).
maxRefreshDeadline	3600	
numRefreshThreads	4	The number of threads in the pool that continuously look at the most urgent deadlines in the queue and perform the refreshes (Elasticsearch queries) as needed.
maxRefreshRetries	3	The number of times a refresh can fail due to a failing Elasticsearch query before an error is logged and the refresh is dropped.
maxQueueSize	10000	Maximum queue size for scope quota evaluations. The internal implementation uses a deadline queue and, if the queue is overflowed, the least urgent items are pushed out of the queue.

queryTTL	maxRefreshDeadline	This avoids unnecessary load on Elasticsearch by allowing the results of a quota check performed when a scope (tenant, domain, bucket) is accessed to be cached for this period of time. If the time since last access is less than this value, the scope is not scanned in the background. Setting this parameter to 0 disables the access caching function.
refreshRetryDelay	10	Number of seconds to wait before retrying a refresh after the previous failed due to a failing Elasticsearch query.
refreshIdleSleep	3	Seconds to wait after finishing the work in a queue and before starting again.
smtpHost	localhost	Required. The hostname or IP address of the SMTP server that sends the email notifications.
smtpPort	25	Optional. The port where the SMTP server listens.
smtpUser smtpPassword		Optional. The user and password to authenticate with SMTP server.
mailFrom	donotreply@localhost	Email address for the sender of the notification.
mailSubjectTemplate	Quota state change notification	Email templates for subject line and body. These variables can be used in both the subject line and message body templates. <ul style="list-style-type: none"> • %metric% • %state% • %contextType% • %contextName% The %xxx% strings render current values when the message is generated.
mailTemplate	Metric %metric% changed to %state% state in %contextType% %contextName%.	

See [Setting Quotas](#).

[dynamic_features]

Any configuration settings appear in this **Dynamic Features** section if optional, dynamic features such as [Video Clipping for Partial File Restore](#) are installed (v11.0).

resultObjectLifetime	5	In days. Sets a lifepoint to trigger clean up of any JSON result objects for video clips are created asynchronously.
-----------------------------	---	--

Setting Ports for Docker or Proxies

Gateway manages communications through assigned ports. Gateway can be configured to run either within a Docker environment or behind a proxy as of release 5.4. The configuration has two settings (`externalHTTPPort`, `externalHTTPSPort`) per protocol: `[scsp]` and `[cluster_admin]`, the [Service Proxy](#). These settings take effect when `X-Forwarded-Proto` appears on the request.

SCSP, S3, and Service Proxy requests must each route to the correct port. Browser requests must use the correct port:

Content UI	<code>/_admin/portal</code>	SCSP port	<code>[scsp]</code>
Swarm UI	<code>/_admin/storage</code>	Service Proxy port	<code>[cluster_admin]</code>

Gateway can redirect users if they attempt to access a UI on the wrong port; to accomplish this,

- the load balancer must set `X-Forwarded-` headers, which Gateway uses to determine which port to use
- `externalHTTP[S]Port` must be configured correctly in `gateway.cfg`

Example load balancer setup	Example settings in gateway.cfg
<p>If an HAProxy load balancer at haproxy.example.com is proxying requests for SCSP and S3 (on a shared port) and for Service Proxy:</p> <pre> frontend www-http bind 0.0.0.0:80 reqadd X-Forwarded-Proto:\ http # Use for [scsp] externalHTTPPort reqadd X-Forwarded-Port:\ 80 default_backend www-backend-scsp ... use_backend www-backend-s3 if iss3 ... frontend www-http-svc bind 0.0.0.0:91 reqadd X-Forwarded-Proto:\ http # Use for [cluster_admin] externalHTTPPort reqadd X-Forwarded-Port:\ 91 default_backend www-backend-svc ... frontend www-https bind 0.0.0.0:443 ssl crt /var/certs/server.pem reqadd X-Forwarded-Proto:\ https # Use for [scsp] externalHTTPSPort reqadd X-Forwarded-Port:\ 443 default_backend www-backend-scsp ... use_backend www-backend-s3 if iss3 ... frontend www-https-svc bind 0.0.0.0:1443 ssl crt /var/certs/server.pem reqadd X-Forwarded-Proto:\ https # Use for [cluster_admin] externalHTTPSPort reqadd X-Forwarded-Port:\ 1443 default_backend www-backend-svc </pre>	<p>...then expose both HTTP and HTTPS in these sections:</p> <pre> [scsp] bindPort = 80 externalHTTPSPort = 443 externalHTTPPort = 80 [cluster_admin] bindPort = 91 externalHTTPSPort = 1443 externalHTTPPort = 91 </pre>

Redirection – This is how redirection is achieved given the example above. A user incorrectly attempts to access `/_admin/storage` on the SCSP/S3 port exposed by HAProxy:

```
https://haproxy.example.com/_admin/storage # default port 443
```

HAProxy proxies this request to Gateway's SCSP port as:

```
GET /_admin/storage
Host: gateway:80
X-Forwarded-Host: haproxy.example.com
X-Forwarded-Protocol: https
X-Forwarded-Port: 443
```

Gateway SCSP knows that it does not handle `/_admin/storage` requests and that `/_admin/storage` is handled by the `[cluster_admin]` port, so it responds with a redirect to the `[cluster_admin]` `externalHTTPSPort` (because `X-Forwarded-Protocol` specifies `HTTPS`; otherwise, it uses `externalHTTPPort`).

```
HTTP/1.1 308 Permanent Redirect
Gateway-Protocol: scsp
Location: https://haproxy.example.com:1443/_admin/storage
```

Enabling the Service Proxy

For most implementations, one Gateway is dedicated to run as [Service Proxy](#) to support cluster administration (via Swarm UI and Management API), and a pool of additional Gateways handle all content management at scale. For test or lightly used clusters, both cluster administration and content management can be enabled on a single Gateway instance.

On the Gateway instance that runs as Service Proxy, make the following changes to its configuration (`gateway.cfg` file):



Important

All these settings are required. Add the `[cluster_admin]` section if the existing configuration file does not include it.

[cluster_admin]	<code>enabled=true</code>	Enables the Service Proxy functionality.
	<code>bindAddress=<IP hostname></code>	Specifies the IP address or host name where Service Proxy listens for incoming storage cluster management API and Metering Query requests.
	<code>bindPort=91</code>	Specifies the port where Service Proxy listens. By convention, this is port 91.
	<code>externalHTTPPort=<port></code> <code>externalHTTPSPort=<port></code>	Optional, one or both. Allows Gateway to be used either behind a proxy or within a Docker environment, taking effect when <code>X-Forwarded-Proto</code> is found on the request. (Gateway uses <code>X-Forwarded-Proto</code> to determine which port to use.) (v5.4)
	<code>platformHost=<IP hostname></code> <code>platformPort=<port></code>	Required for Platform Server if running Service Proxy /Swarm UI on a standalone Gateway. See Legacy Configuring Swarm for Platform Server .
	<code>secretKey=<key></code>	Set <code>secretKey</code> so the Service Proxy's link obfuscation is consistent across Gateways if using multiple Gateways (running on dual CSNs for HA/load-balancing). A random key is generated when Gateway is started, causing open browsers to break (screen fills with red boxes) until the browser itself is closed and reopened if this is not set.
	<code>testMode=<true false></code>	Enable <code>testMode</code> when troubleshooting, which stops obfuscation of the backend Swarm Storage and Elasticsearch node IPs.

[storage_cluster]	<code>managementPort=91</code>	Specifies the port where Swarm listens for storage cluster management API requests. By convention, this is port 91.
	<code>managementUser=<Swarm.admin.user></code>	Specifies the user known to Swarm allowed to perform management API requests against the storage cluster.
	<code>managementPassword=<Swarm.admin.password></code>	Specifies the password of the <code>managementUser</code> .
[s3]	<code>enabled=false</code>	<p><i>Important:</i> Retain the [s3] and [scsp] sections and explicitly set <code>enabled=</code>, rather than delete them.</p> <p><i>Tip:</i> For test or lightly used clusters, enable one or both protocols on a single Gateway instance configured as the Service Proxy.</p>
[scsp]	<code>enabled=false</code>	

Authentication and authorization for the Service Proxy uses Content Gateway's root IDSYS and root Policy. The root Policy *must* grant all actions to the storage administrator users and/or groups:

```
{
  "Version": "2008-10-17",
  "Statement": [{
    "Action": ["*"],
    "Resource": "*",
    "Effect": "Allow",
    "Principal": {"user": ["admin", "admin2"]},
    "Sid": "storage-admins"
  }],
  "Id": "id-170428899"
}
```

See [IDSYS Document Format](#) and [Policy Document](#).

Gateway Requirements

- [Prerequisites](#)
- [System Requirements](#)
 - [Do not install on management node](#)
- [Space Requirements](#)
 - [Spool space for multipart uploads](#)
 - [Best practice](#)
 - [Logging space](#)
- [S3 Requirements](#)

The Gateway service is essentially a reverse proxy with some protocol inspection duties. As a proxy between the client applications and the storage nodes, its primary duty is to pass bytes from one network adapter to another.

Prerequisites

Content Gateway requires the following components for installation and operation:

- Swarm Storage cluster implemented with [Storage settings needed by Gateway](#)
- [Elasticsearch cluster installed](#) (if using metering data for critical functions such as billing, deploy at least 3 Elasticsearch servers and [set up snapshot backups](#) of that data)
- [Search feed defined and enabled](#)
- An [authentication backend](#), such as LDAP or PAM
- Network time protocol (NTP) server reachable by both Gateway server(s) and Swarm storage nodes
- At least one server on which to install the Content Gateway software

System Requirements

The system requirements for the Gateway depend on the volume of the traffic and the speed of the upstream network connection to client applications.



Do not install on management node

Content Gateway and the production Elasticsearch cluster need to be on separate machines from the management node (Platform Server or CSN). The management node installs with Service Proxy and a single-node ES, which are dedicated to the Swarm UI.

- Gateway server software:
 - 64-bit Linux operating system, RHEL or CentOS 7
 - Java 8 (earlier and later versions of Java are not supported)
- Gateway server hardware:
 - Virtual or physical machine
 - 2+ CPU cores
 - 2+ GB RAM
 - 3+ GB /tmp space

- 2+ GB available disk storage after OS installation (see *Space Requirements* below)
- For high availability and capacity scaling, add the following:
 - Two or more additional Gateways
 - A load-balancing mechanism
- Prevent Gateway clients from making storage requests directly to the back-end storage cluster using one of these methods:
 - (*most common*) Make the Gateway servers dual-homed on the front-end client network and the back-end storage network.
 - Use network filtering to prevent direct user access to the storage cluster and to deploy Gateway servers and storage servers on one subnet.
 - Use VLAN tagging on the Gateway server's network interface to allow one physical interface to carry both front-end and back-end traffic.

Space Requirements

Spool space for multipart uploads

The HTTP multipart MIME upload operation requires spool space on the Gateway server; all other operations, including the S3 multipart upload, SCSP multipart writes, and normal whole-object writes, stream through the Gateway and directly to the back-end object storage nodes. HTTP multipart MIME POST requests are used by the upload function in the Content Portal and by HTML form POSTs.

Verify the total free disk space on a Gateway server includes an allowance for the maximum expected to be needed for these requests. To control the spool location and the percentage of disk space that can be used, set the `multipartSpoolDir` and `multipartUsageAllowed` in the `[gateway]` section of the configuration file.



Best practice

If the [Content UI Overview](#) has many users and/or large uploads, increase the available space in the Gateway's spool directory to 32 GB or more and increase the `multipartUsageAllowed` parameter value.

Logging space

The Gateway server uses up to 2GB of disk space for application logs and audit logs in the default configuration. The retention time and file size of the historical logs can be changed as required based on the deployment requirements. See [Logging Configuration in the Gateway configuration](#) section.

S3 Requirements

Follow these requirements to use S3 with Content Gateway:

- Enable and configure erasure-coding (EC).
- Size the cluster to support EC; for example, do not attempt to use S3 with inadequate resources, such as 3 chassis and `reps=2`.

See [Configuring Swarm Storage for Gateway](#) and also [Erasure Coding EC](#) and [Hardware Setup](#) in the Swarm Storage guide.

Configuring Swarm Storage for Gateway

This section provides information specific to running Swarm Storage with Gateway. Install and configure Swarm, the storage cluster (storage nodes running on dedicated hardware) before proceeding.

- [Network Placement](#)
 - [Caution](#)
- [Domain Management](#)
- [Elasticsearch Servers](#)
 - [Listing consistency](#)
- [Configuration Requirements](#)
 - [Caution](#)
 - [Note](#)

Network Placement

When deployed with Gateway, the storage nodes should be placed on a network subnet not directly accessible to client applications. All user communications with the storage cluster must go through the Gateway.



Caution

If users are allowed to communicate directly with the storage cluster nodes, they may bypass access security, the business rules for content metadata, and audit logging performed by the Gateway and may render content in the cluster unusable to the Gateway. Only allow direct access to the storage cluster nodes under highly controlled circumstances, such as administrator-only operations or trusted applications.

Domain Management

The Swarm cluster provides for logical separation of content among multiple tenants through the use of storage domain names. Gateway has the following requirements beyond those for a baseline storage deployment and client usage.

- An administrative domain must be created in the storage cluster.
- Storage domains must adhere to IANA naming standards (valid DNS names).
- Client applications should specify a storage domain in every request (if not, the request goes to the default domain, with `enforceTenancy=True`).

The storage domain name for an operation is specified by the client application according to the following precedence from highest to lowest:

- SCSP `domain=X` query argument
- HTTP `X-Forwarded-Host` header
- HTTP `Host` header

Storage domains in Swarm must resolve to at least one IP address ("A" record) for client applications to make use of the Host header to identify the storage domain with most HTTP/1.1 libraries. The resolved IP address should be for a Gateway or some other front-end network appliance such as a load balancer if applicable. Using a DNS round-robin with IP addresses is a valid configuration to use if there are multiple Gateway servers.

This is an example of a *BIND 9* zone file implementing a wildcard of all storage domains within the `cloud.example.com` parent DNS domain and points them to the IP address `10.100.100.100`.

```
$TTL 600 @ IN SOA cloud.example.com. dnsadmin.example.com. (
  2016070201 ; Serial number
  4H      ; Refresh every 4 hours
  1H      ; Retry every hour
  2W      ; Expire after 2 weeks
  300 ) ; nxdomain negative cache time of 5 minutes
IN NS ns1.example.com.
* IN A 10.100.100.100
```

In the example zone file, `10.100.100.100` is the IP address used by client applications to communicate with the Gateway or a front-end load balancer. The names hydrogen2.cloud.example.com and oxygen.cloud.example.com both resolve to the same IP address.

Elasticsearch Servers

When using the S3 storage protocol, the metadata search service must be accessible to the Gateway servers.

When deployed with Gateway, like the storage nodes, the typical placement is on a network subnet not directly accessible to the client applications. There are no end-user supported API calls directly to the metadata search service.

Listing consistency

Search feeds show eventual consistency as content changes, but enabling the [Gateway Configuration \[s3\]](#) option `enhancedListingConsistency` improves the search-after-create response to the client applications using the Gateway.

Configuration Requirements

Use these Swarm configuration settings and adhere to the following operational changes when using Swarm Storage with Gateway. These configuration changes refer to the configuration file(s) for Swarm.

- **CSN** – This is the cluster-wide file: `/var/opt/caringo/netboot/content/cluster.cfg`
- **Platform Server** – This is the cluster-wide file used to deploy, which is located by default here: `/etc/caringo/cluster.cfg`
- **No platform server** – This is the node-specific configuration file: `node.cfg`

Caution

Failure to use these settings and operational changes can prevent Gateway from working properly with the storage cluster.

Requirement	Description
Optimize GETs	<p>With Swarm 12.0 and higher, a setting can be added to improve performance through Gateway. Enable <code>scsp.enableVolumeRedirects</code> to permit Gateway to redirect GET requests to volume processes. These redirects increase efficiency, especially with reading small objects.</p> <pre>scsp.enableVolumeRedirects = True</pre>

<p>Enable an EC encoding</p>	<p>S3 multipart (large file) writes fail if erasure coding is not configured; define an ecEncoding if using S3.</p> <pre>policy.ecEncoding = {k:p}</pre> <p>See Implementing EC Encoding Policy.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>the value put into the configuration file is used on first boot. After the cluster is running, the original values are persisted and must be updated dynamically (using UI or SNMP). See Persisted Settings (SNMP).</p> </div>
<p>Clear legacy settings</p>	<p><i>Unless needed for backwards compatibility</i> (because untenanted objects are used in the cluster and do not need S3), enable tenancy for unnamed objects, which verifies every object is written to a domain (see How enforceTenancy Works):</p> <pre>cluster.enforceTenancy = True</pre> <p>Set it to <code>True</code> and reboot the cluster if this was set to <code>False</code>.</p>
<p>Storage Domain Management</p>	<p>Only create and manage storage domains through the Content UI or programmatically through the Gateway's management API.</p> <p>The cluster configuration contains <code>security.noauth=False</code> if storage domain management displays in the legacy Admin Console (port 90), which is not supported by Content Gateway. Set it to <code>True</code> and reboot the cluster.</p> <p>Troubleshooting: If the Content UI reports "Page Not Found: The original bucket to which this collection refers cannot be found or has been replaced", it is likely the domain was created by the legacy Admin Console (port 90) and contains the legacy <code>Castor-Authorization</code> header. DataCore Support for help correcting the domain.</p>

Deployment Process

Following is a high-level view of the nature and order of tasks that need to be performed for a full-stack Swarm implementation.



Important

Complete [Deployment Planning](#) in consultation with DataCore Sales and Support before starting these tasks.

- [Important](#)
- [Phase 1: Prepare Environment](#)
- [Phase 2: Platform Server and Storage Cluster](#)
- [Phase 3: Elasticsearch](#)
- [Phase 4: Content Gateway](#)
- [Phase 5: Swarm Clients](#)
 - [Optional Swarm Components](#)
- [Phase 6: Post-installation](#)

Phase 1: Prepare Environment

The work to prepare the environment must be completed *before* adding any Swarm components:

- Rack and stack hardware designated for Swarm, replacing and upgrading as needed. (See [Hardware Setup](#).)
- Upgrade firmware to latest versions:
 - All servers
 - All disk controllers
 - All disk drives
- Configure networking and switches (see [Network Infrastructure](#)), including the following:
 - VLAN configuration
 - IGMP snooping disabled (or IGMP querier implemented)
- Configure IPMI management.
- Provide access for the storage cluster to phone home. (See [Cluster Health Report](#).)
- Verify the servers and base operating systems meet the Swarm system requirements. (See [Hardware Requirements for Storage](#) and [Hardware Requirements for Elasticsearch Cluster](#).)
- Configure IPMI (remote server management)
- Complete licenses and agreements
 - Obtain any needed storage capacity and capability licenses from DataCore. (See [Licensing Swarm](#).)
 - A user in the organization must register the Red Hat license and accept the EULA if installing or updating RHEL.
 - Accept the [DataCore EULA](#).

Phase 2: Platform Server and Storage Cluster

Swarm Platform Server is installed and configured first, so it can install Storage nodes on the designated hardware.

- Install Platform Server. (See [Legacy Platform Implementation](#).)

- Configure Platform Server to integrate with the environment.
- Configure Platform Server to boot the current version of Swarm Storage.
- Boot the Storage nodes and configure the cluster-wide settings. (See [Configuring Swarm Storage.](#))
- Install the Swarm Storage UI. (See [Swarm Storage UI Installation.](#))
- Verify the storage cluster is operational: read, write, and delete test objects
- *Optional:* Install the open-source components to make use of Swarm's exports to Prometheus. (See [Prometheus Node Exporter and Grafana](#) .)

Phase 3: Elasticsearch

Install and configure an Elasticsearch cluster on designated hardware, providing the Storage cluster the search and metrics capabilities.

- Base install the chosen operating system (RHEL/Centos 7.9).
- Install Elasticsearch nodes on designated hardware. (See [Elasticsearch Implementation.](#))
- Configure Elasticsearch based on DataCore recommendations. (See [Configuring Elasticsearch.](#))
- Create a Search feed to populate the Elasticsearch metadata index. (See [Managing Feeds.](#))
- Configure Elasticsearch Curator and Swarm Metrics. (See [Installing Swarm Metrics.](#))

Phase 4: Content Gateway

Install and configure Content Gateways, which provide the primary access to the Storage cluster.

- Base install the chosen operating system (RHEL/Centos 7.9).
- Install the Content Gateway. (See [Content Gateway Implementation.](#))
- Install the Content UI. (See [Content UI Installation.](#))
- Configure basic gateway setup for verification and initialization of primary domain. (See [Gateway Configuration](#) and [Configuring Swarm Storage for Gateway.](#))
- Verify the Gateway is operational: read, write, and delete test objects using the Content UI, S3, and SCSP.
- Create the initial domains, with policy definitions. (See [Gateway Access Control Policies.](#))

Phase 5: Swarm Clients

As fits with the implementation plan, extend access to Swarm storage by installing one or more Swarm client applications, such as the following:

- [SwarmFS Implementation](#)
- [FileFly](#)



Optional Swarm Components

These are optional Swarm components, each with separate distribution packaging and licensing.

Phase 6: Post-installation

- Conduct performance measurement and tuning.
- Test and debug third-party/custom applications and integration.
- Train administrators and staff.

Network Infrastructure

This section describes how to set up your Swarm network infrastructure in your corporate enterprise.

- [Setting up PXE Booting](#)
- [Network Devices and Priority](#)
- [Setting up the Network Services](#)
- [Understanding Swarm in the Network](#)
- [Tuning Network Performance](#)
- [Proxying the Swarm Admin Console](#)
- [Setting up the Swarm Network](#)
- [IGMP Snooping](#)

Setting up PXE Booting

- [Setting up the DHCP server for PXE booting](#)
- [Configuring PortFast on switch ports](#)
- [Configuring the TFTP server](#)
- [Setting up a configuration file server](#)
- [Disabling monitor power-saving activation](#)

This section describes how to boot a cluster from the network using the Intel [Preboot Execution Environment](#) (PXE) specification. This booting process (commonly referred to as *network booting*) is supported by most NICs. PXE is one way to boot the storage cluster nodes.

i **Platform Server**

If using [Platform Server](#), *skip this section*: the network booting is set up.

- To enable nodes to boot from a USB flash drive, see [Initializing a Storage Cluster](#).
- To enable nodes to boot using a configuration file server, see the section below.
- To enable nodes to PXE boot, perform these steps:
 1. Configure the DHCP server with next-server and filename parameters.
 2. Configure PortFast on the switch ports leading to the storage cluster nodes.
 3. Configure the TFTP server with PXE bootstrap, configuration, and Swarm files.
 4. Set up the nodes' BIOS configurations for network booting.

i **Requirement**

Increase the size of the initrd RAM disk to 160MB on the PXE boot server to prevent PXE boot failures. This does not apply if using Platform Server.

Setting up the DHCP server for PXE booting

Warning

Swarm can erase all non-Swarm data on hosts that boot accidentally from the network. When setting up the DHCP server, verify it provides network booting information to the correct network hosts.

The following example shows the configuration lines from the Internet Systems Consortium (ISC) DHCP server that is commonly available on UNIX systems. The `next-server` parameter defines the IP address of the Trivial File Transfer Protocol (TFTP) server and the `filename` parameter to define the bootstrap loader program to download as shown below:

```
group {
  next-server 172.16.1.10;
  filename "/pxelinux.0";
  # Hosts allowed to network boot into Swarm
  host clusternode1 { hardware ethernet 00:90:cb:bf:45:26; }
  host clusternode2 { hardware ethernet 00:90:b2:92:09:e4; }
  host clusternode3 { hardware ethernet 00:90:0d:46:7a:b4; }
}
```

The Swarm nodes are explicitly defined by MAC address to prevent Swarm from initiating an unattended boot by other servers or workstations in this example.

Configuring PortFast on switch ports

PortFast is a switch port configuration parameter that enables a port to bypass the listening and learning [Spanning Tree](#) states so the port immediately forwards traffic.

Verify PortFast is configured on the switch ports leading to each node if a storage cluster node is connected to a network switch. The extended time delay can prevent netboot from delivering the Swarm image to a PXE-enabled node in a timely manner if this condition is not met.

Configuring the TFTP server

The TFTP server transfers configuration or boot files between systems in a local environment. Configure the TFTP server to load the Swarm software onto the cluster nodes after configuring the DHCP server.

To set up the TFTP server:

1. Install and configure TFTP server software on the boot server.
2. Create the `/tftpboot` directory hierarchy.
3. Copy the kernel and fsimage files to the `/tftpboot/profiles/castor` directory.

See [DHCP and Boot Server Redundancy](#), below.

Installing and configuring TFTP

TFTP server software is available in both free and commercial packages. UNIX distributions commonly include TFTP server software with the standard setup. The `tftp-hpa` package for UNIX can integrate with Swarm. Source code can be obtained from the Linux Kernel Archives website located at kernel.org/pub/software/network/tftp.

TFTP server software is also available as a binary package in many Linux distributions.

Creating the tftpboot directory hierarchy

Configure the server to access the network boot file directory after installing the TFTP server. This directory is typically labeled `/tftpboot` because TFTP is almost exclusively used for booting network devices.

A sample template is included in the `samples/NetworkBoot` directory of the Swarm software distribution.

Copying kernel and fsimage

The Swarm software distribution media includes the kernel and fsimage files, which contain the Swarm embedded operating system. Copy these files to the `tftpboot/profiles/castor` directory on the TFTP server so they load onto each Swarm node during bootup.

The `tftpboot` directory on the TFTP server should contain these files after copying the directory template and the Swarm software files:

File name	Description
<code>tftpboot/pxelinux.0</code>	Boot loader program
<code>tftpboot/profiles/castor/fsimage</code>	Swarm software
<code>tftpboot/profiles/castor/kernel</code>	Swarm operating system kernel
<code>tftpboot/pxelinux.cfg/default</code>	PXELINUX configuration file

See the documentation and ZIP file in the `samples/Network-Boot` directory on the Swarm distribution media for help with using the PXELINUX boot loader.

DHCP and boot server redundancy

Configure both a primary and secondary DHCP server when setting up the DHCP server. This configuration eliminates a single point of failure if one of the servers goes offline for any reason.

- See "Failover with ISC DHCP" at madboa.com/geek/dhcp-failover to set up the ISC DHCP daemon for redundancy.
- Use the primary and secondary DHCP servers as TFTP servers to provide redundancy at the network booting layer. Set the next-server parameter in each server to specify a separate IP address when setting up the DHCP servers. the primary or secondary DHCP server handles the PXE boot when it answers a DHCP query.
- Verify the TFTP boot servers are located in the same broadcast domain (or VLAN) as the Swarm nodes or enable a DHCP relay server on the VLAN to prevent any network interruptions.

Setting up a configuration file server

Platform Server

Skip this section if using [Platform Server](#): the centralized configuration is set up.

Swarm supports centralized node configuration files on an HTTP or FTP server. This method allows booting from a network or a standard USB flash drive. A centralized configuration file server simplifies storage cluster administration by supporting configuration file updates and providing a method to group similar node configurations together.

Set the value of the **castor_cfg** kernel configuration parameter to a URL that targets the configuration list file to implement a configuration file server as described below.

PXE boot example

This is an example PXELINUX configuration file located in the **tftpboot/pxelinux.cfg** directory on the TFTP boot server.

```
default profiles/castor/kernel
append initrd=profiles/castor/fsimage ramdisk_size=160000 root=/dev/ram0
    castor_cfg=http://172.16.1.200/castor/cfg-list.txt
```

USB boot loader example

This is an example section of the `syslinux.cfg` located in the root directory on the USB flash drive:

```
label normal
    kernel kernel
    append initrd=fsimage ramdisk_size=160000 root=/dev/ram0
        castor_cfg=http://172.16.1.200/castor/cfg-list.txt
```

Configuration list file example

The **castor_cfg** kernel configuration parameter specifies a file containing a list of URLs for all configuration files to be loaded by a Swarm node. Swarm configuration files are evaluated in the order in which they are listed in the configuration list file.

Although Swarm configuration settings can be defined multiple times, the last definition is used. By redefining the settings, configuration files can be layered so they contain generally applicable values for a cluster, a group of similar nodes, and values specific to one node.

Example of URLs in a configuration list file:

```
http://172.16.1.200/castor/cluster.cfg
http://172.16.1.200/castor/subcluster.cfg
http://172.16.1.200/castor/testnode.cfg
```

Each of the configuration files in the list file uses the same format as the `Swarm node.cfg` file.

See the **/caringo/node.cfg.sample** in the Swarm software distribution.

See [Managing Configuration Settings](#).

Disabling monitor power-saving activation

Add the following kernel option to the APPEND line in either the **syslinux.cfg** file on the Swarm USB key or in the PXE boot configuration file to disable the monitor power-saving feature from activating while connected to a Swarm storage node.

This parameter tells the kernel to stop blanking the console when enabled:

```
consoleblank=0
```

This feature defaults to 10 minutes. A value of 0 disables the blank timer. Listed below are examples:

PXE boot example

This is a PXELINUX configuration file from the **tftpboot/pxelinux.cfg** directory on the TFTP boot server with the console power saver disabled.

```
default profiles/castor/kernel
append initrd=profiles/castor/fsimage consoleblank=0 ramdisk_size=160000 root=/dev/ram0
castor_cfg=http://172.16.1.200/castor/cfg-list.txt
```

USB boot loader example

This is a section of the **syslinux.cfg** contained in the root directory on the USB flash drive with the console power savings disabled.

```
label normal
kernel kernel
append initrd=fsimage consoleblank=0 ramdisk_size=160000 root=/dev/ram0
castor_cfg=http://172.16.1.200/castor/cfg-list.txt
```

Network Devices and Priority

By default, all Swarm Ethernet network adapters are encapsulated into a redundant bond interface where one device is active and the remaining devices are backups. In this default mode, the first network device is the preferred device. Override this behavior by:

- Excluding a network adapter from use, such as an [intelligent platform management interface](#) (IPMI) card.
- Changing the preferred network adapter for network load management.
- Bonding multiple adapters together for increased throughput (such as *NIC teaming*).

To override the network adapter, configure the switch ports to the appropriate mode. Swarm supports these Linux bonding driver modes:

- **active-backup** (Active-backup)
- **balance-alb** (Adaptive load balancing)
- **802.3ad** (IEEE 802.3ad)

To override the Swarm default network device settings, edit one of the following boot configuration files:

- **syslinux.cfg** if the node is booting from the USB flash drive
- **pxelinux.cfg** if the node is booting from the network

In the configuration file, a kernel parameter named **castor_net** is included in the `append` clause. **castor_net** allows specifying both the bonding mode for the adapters as well as a comma-separated ordered list of the network devices Swarm can use. The first device in the list is the preferred interface used when online.

Important

The list of network devices *must* use the adapter names assigned by Swarm. To locate the current list of adapter names and MAC addresses, boot the node from the Swarm USB drive to access the [System Menu](#).

Swarm assigns device names to adapters based on a sorted list of MAC addresses. Adding network hardware can change the assignment order.

Below are some examples of assigning device names in the configuration file. The other portions of the `append` clause are abbreviated for clarity. Note the trailing colon after the bonding mode:

```
append initrd=... castor_net=active-backup:eth1,eth0
append initrd=... castor_net=balance-alb:eth0,eth1
append initrd=... castor_net=802.3ad:eth1,eth0
append initrd=... castor_net=eth1,eth2
append initrd=... castor_net=802.3ad:
```

Setting up the Network Services

This section describes how to set up the network services for a storage cluster.



Platform Server

Skip this section if using [Platform Server](#): network services are set up.

- [Platform Server](#)
- [Setting up NTP for time synchronization](#)
 - [NTP 3.0](#)
- [Setting up DHCP for IP address administration](#)
- [Setting up DNS for name resolution](#)
 - [Best practice](#)
 - [Best practice](#)
- [Preparing for domains](#)
- [Setting up a Syslog Server for Critical Alerts](#)
- [Setting up SNMP for monitoring](#)
 - [Disabling SNMP](#)
- [Setting up network load balancing](#)
- [Setting up the network interfaces](#)

Setting up NTP for time synchronization

The Network Time Protocol (NTP) server provides time synchronization between the cluster nodes, which is critical for many Swarm components. For best results, configure multiple NTP servers in close proximity to a cluster. For example, use the [NTP Pool Project's continental zones](#), which are pools of NTP servers.

One or more trusted NTP servers, such as dedicated hardware solutions on the internal network or publicly available NTP servers, are required in a storage cluster. This configuration is required, verifying the internal clocks in all nodes are synchronized with each other.

Add trusted NTP servers to the cluster by adding the IP addresses or host names in the `network.timeSource` parameter located in the node configuration files if available. The parameter value is a list of one or more NTP servers (either host names or IP addresses) separated by spaces. For example, to add a second NTP server IP address, use the following syntax:

```
network.timeSource = 10.20.40.21 10.20.50.31
```

The node must be able to resolve host names using a DNS server to add an NTP server host name. Use this syntax:

```
network.timeSource = ntp1.example.com ntp2.example.com
```

See [Configuring an External Time Server](#).



NTP 3.0

NTP 3.0 included a design limitation causing the time value to wrap in the year 2036. NTP cannot correct the time if the BIOS clock in a cluster node is set beyond this wrap point. Verify the BIOS clocks in all nodes are set to a year prior to 2036 before booting Swarm in a cluster. This issue was resolved in NTP 4.0.

The node does not boot if the configured NTP server(s) cannot be reached. See [Configuring a Node without NTP](#) if the cluster nodes cannot access an external or internal NTP server.

Setting up DHCP for IP address administration

The Dynamic Host Configuration Protocol (DHCP) server provides IP addresses to the cluster nodes and other devices enabled as DHCP clients. While Swarm nodes are not required to have static IP addresses to discover and communicate with each other, administrators may find it easier to manage and monitor a cluster where each node receives a predetermined IP address.

Configure this option using DHCP:

1. Map the Ethernet media access control (MAC) address of each node to a static IP address.
2. Configure the DHCP server to provide *each node* with an IP address for each of these:
 - network mask
 - default gateway
 - DNS server

Setting up DNS for name resolution

The Domain Name Service (DNS) is used to resolve host names into IP addresses. While DNS is not required for Swarm nodes to communicate with each other, DNS can be very useful for client applications to reach the cluster. DNS is one method to use to enable access to objects over the Internet if using named objects.



Best practice

Although client applications can initiate first contact with any node in the storage cluster – even choosing to access the same node every time – best practice is for the node of first contact to be distributed evenly around the cluster.

For example:

- Define multiple DNS entries ("A" or "CNAME" records) that specify the IP address for the same Swarm first contact node.
- Use multiple IP addresses for a DNS entry to create a DNS round-robin that provides client request balancing.

See the DNS software documentation for how to use "A" records and "CNAME" (alias) records.

Swarm requires a DNS server to resolve host names in the configuration file. For example, add a host name to the NTP list or the log host (such as ntp.pool.org) for name resolution. The DNS server needs to be set in the Swarm configuration file. In contrast, applications must resolve Swarm domain names to find the storage cluster. These unique requirements are most likely addressed using different DNS servers.

The following example shows the entries in the Internet Systems Consortium (ISC) BIND DNS software configuration file for three node IP addresses tied to one name.

```
Swarm 0 IN A 192.168.1.101
      0 IN A 192.168.1.102
      0 IN A 192.168.1.103
```

In this example, the [Time To Live](#) (TTL) value for each of the records in the round-robin group is very small (0-2 seconds). This configuration is necessary so clients who cache the resolution results quickly flush them. This process allows the first contact node to be distributed and allows a client to move quickly to another node if the first contact node is unavailable.



Best practice

Applications are recommended to implement robust mechanisms such as [Zero Configuration Networking](#) for distributing the node of first contact and skipping failed nodes. An administrator can use DNS to assist with less complex applications.

Preparing for domains

To allow clients to access named objects over the Internet, enable incoming HTTP requests to resolve to the correct domain. (A cluster can contain many *domains*, each of which can contain many *buckets*, each of which can contain many named objects.) Cluster and domain names should both be Internet Assigned Numbers Authority (IANA) compatible host names, such as `cluster.example.com`.

For example, a client application can create an object with a name such as:

```
cluster.example.com/marketing/photos/ads/object-naming.3gp
```

In this example, `cluster.example.com` is the domain name, `marketing` is the name of a bucket, and `photos/ads/object-naming.3gp` is the name of an object. Set up the network so the host name in the HTTP request maps correctly to the object's domain name. The cluster name is not required.

To enable clients to access a named object:

1. Set up host files to map domain names to IP address(es) of the first contact node.
 - For a Linux system, configure the `/etc/hosts` file.
 - For a Windows system, configure the `%SystemRoot%\system32\drivers\etc\hosts` file.

```
192.168.1.111 cluster.example.com
192.168.1.112 vault.example.com
```

2. Define multiple DNS entries ("A" or "CNAME" records) that identify the IP address(es) of the first contact node in the storage cluster. This process creates a DNS round-robin that provides client request load balancing.
 - For help setting up DNS for Swarm, see *Setting up DNS for name resolution*, above.
 - For information about setting up a DNS server, see the DNS software documentation.

Setting up a Syslog Server for Critical Alerts

A syslog server must be set up to capture critical operational alerts from the nodes in a storage cluster. The server captures messages sent by the Swarm nodes on UDP port 514.

See [Configuring External Logging](#) on configuring an rsyslog server and the log.host and log.level parameters used to send Swarm messages to a syslog server.

Setting up SNMP for monitoring

Swarm provides monitoring information and administrative controls using the Simple Network Management Protocol (SNMP). Using an SNMP console, an administrator can monitor a storage cluster from a central location.

 **Disabling SNMP**

Disable the Swarm Storage setting `snmp.enabled` (v12.0) if SNMP needs to be disabled cluster-wide, such as for a security need or using Swarm in containers.

Swarm uses an SNMP [management information base](#) (MIB) definition file to map SNMP [object identifiers](#) (OIDs) to logical names. The MIB can be located in one of two locations, depending on the configuration:

- The aggregate MIB for the entire cluster is located at `/usr/share/snmp/mibs` if cluster nodes boot from a Platform Server.
- The MIB is located in the root directory of the Swarm software distribution if cluster nodes *do not* boot from a Platform Server.

See [Using SNMP with Swarm](#).

Setting up network load balancing

Although the Swarm Storage Cluster nodes interact with client applications using the HTTP communication protocol, the nodes operate differently from traditional web servers. Placing storage nodes behind an HTTP load balancer is usually an unnecessary configuration. A properly configured load balancer can add value-added services like SSL off-load and centralized certificate management.

During normal operations, a storage node routinely redirects a client to another node within the cluster. The client must initiate another HTTP request to the redirected node when this process occurs. Any process that virtualizes the storage node IP addresses or attempts to control the nodes connected to the client generates communication errors.

Setting up the network interfaces

Gigabit Ethernet or faster NICs provide the recommended 1000 Mbps data communications speed between storage cluster nodes. Swarm automatically uses multiple NICs to provide a redundant network connection.

Connect the NICs to one or more interconnected switches in the same subnet to implement this feature.

See [Switching Hardware](#).

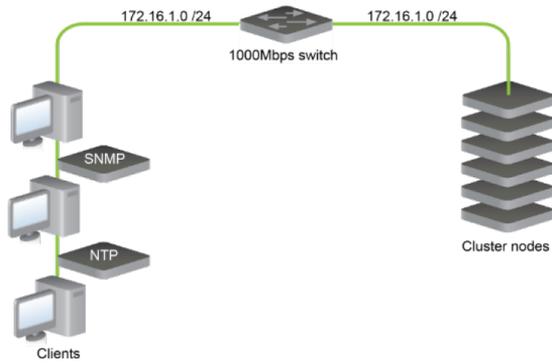
Understanding Swarm in the Network

This section provides a high-level overview of setting up a storage cluster in a network.

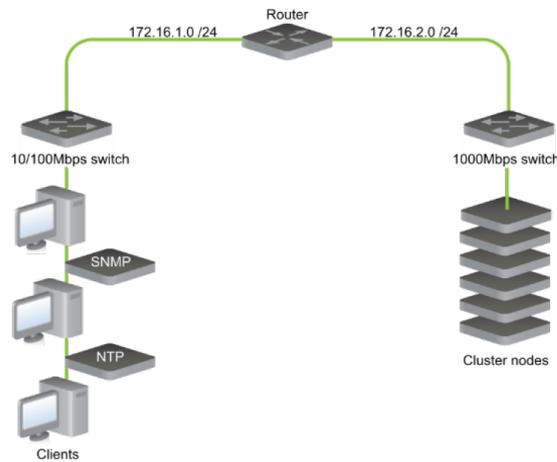
- [Sample Networks](#)
- [Layer 3 Switching and Routing](#)
- [Switching Hardware](#)
- [Internet Deployments](#)

Sample Networks

The following illustration shows a network where the storage cluster nodes and clients are located in the same subnet using a 1000 Mbps switch. This network is easy to set up and requires basic hardware components, but does not offer any traffic separation between the Swarm nodes and the remaining network.



The next illustration shows a network where the storage cluster nodes and clients are located on separate subnets using a router.



Layer 3 Switching and Routing

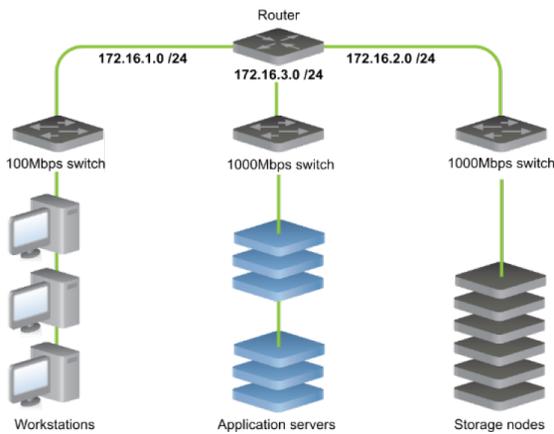
A router or an [Open Systems Interconnection \(OSI\) layer 3](#) switch routes network packets between subnets. A router segregates network traffic by filtering packets based on the targeted subnets. Separating the subnets provides the Swarm nodes with a stable network bandwidth so the multicast and unicast traffic between each node in a storage network does not interfere with the systems and devices in the corporate network.

Switching Hardware

If client workstations are configured with 100 Mbps network interface controllers (NICs) or cannot effectively use more than 100 Mbps of bandwidth, connecting these systems to 1000 Mbps Ethernet switches may not be cost-effective. In this case, consider connecting these workstations to a separate Ethernet switch that supports the slower bandwidth speed.

Many client workstations are configured with 100 Mbps NICs and it may not be cost-effective to connect these workstations to 1000 Mbps ports. The operating systems and applications running on these workstations may be unable to use more than 100 Mbps of bandwidth effectively.

The following network architecture has the client workstations, application servers, and Swarm storage nodes isolated on switches that support the maximum bandwidth speeds.



Using advanced switches supporting multiple routing capabilities, network segments can be isolated as [Virtual LANs](#) (or *VLANs*) on the same device.

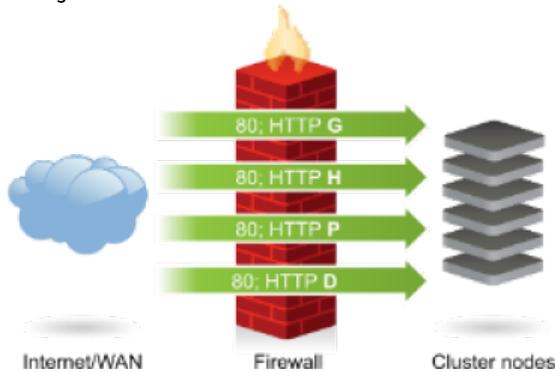
Design the Swarm storage network subnet to incorporate redundant switches to provide high availability when a switch fails. A redundant path provides uninterrupted data communications between the nodes if a switch fails for any reason when Swarm nodes are connected to multiple network switches. Deploying Swarm in a multiple switch environment (or [switched fabric](#)) requires planning and an understanding of the corporate IT structure.

The bandwidth in the switched fabric needs to *exceed* the port speed on each switch to provide effective data communications between each switch port. Contact the switch provider for information about proprietary software or implementing [link aggregation](#) in the Swarm network.

Internet Deployments

When deploying any service on the Internet or within an extensive enterprise wide area network (WAN), network security is a top priority. In these situations, install a firewall or filtering router in front of the storage cluster nodes to control the types of traffic and requests that access the cluster nodes.

The following illustration shows a firewall that allows requests on TCP port 80, the default Simple Content Storage Protocol (SCSP) port. If the SCSP port value set in the storage cluster node or cluster configuration file is not port 80, reset the firewall TCP port to match the value in the configuration file.



Additional configuration is required to allow the supported SCSP methods if the firewall can examine HTTP request content or traffic on [OSI layer 7](#) (the Application layer).

- To present a cluster as a read-only device to external clients, block the POST and DELETE requests to prevent updates to the cluster.
- To prevent client access to the Node Status window in the Swarm Admin Console, configure the firewall to deny "GET /" requests to the cluster nodes.
- To prevent unauthorized access to the Swarm Admin Console, block Internet access to the Swarm Admin Console port (default TCP port 90) and the SNMP port (UDP port 161). Wide area networks (WANs) may require additional restrictions to prevent access to specific administrative networks or workstations.
- To minimize the client impact of hardware failures, deploy devices in redundant pairs when adding security devices such as firewalls into the network architecture.

Tuning Network Performance

- [Important](#)
- [Running Benchmarks](#)
 - [Best practice](#)
 - [Benchmark prep](#)
- [Enable Jumbo Frames](#)
 - [Important](#)
 - [Caution](#)
- [Change sysctl Settings](#)
 - [RHEL 7](#)
 - [Gateway components and clients](#)
 - [Modified /etc/sysctl.conf](#)
 - [Important](#)
 - [Swarm storage nodes](#)
- [Set Buffer Size](#)

Network tuning is beyond the scope of Swarm Storage, but it is an important part of the implementation. Contact DataCore for assistance with network tuning.

By default, Linux networking is configured to optimize reliability, not performance, which becomes apparent with GbE adapters: the kernel's send /receive buffers, TCP memory allocations, and packet backlog are generally too small. With gigabit Ethernet, tuning can significantly improve performance.



Important

For best results, run benchmarks on non-Swarm nodes in the network, using tools such as FTP to perform data transfers to measure performance. After settings are optimized for non-Swarm systems in the network environment, apply those settings to the Swarm nodes in the cluster.

For each GbE controller, Intel includes a README for Linux outlining recommendations. Refer to the documentation supplied by the manufacturer of the controller.

These are the most important performance-tuning changes to make (per Intel's ixgb driver documentation), in order of greatest impact:

1. Enable **jumbo frames** on the local hosts and switches.
2. Use **sysctl** to tune the Linux kernel settings.

See [Performance tuning: Intel 10-gigabit NIC](#), which incorporates the tuning recommendations [cited in the Intel documentation](#).

Running Benchmarks

Best practice

Change one setting at a time, running benchmarking tools (such as `iperf` and `netperf`) to determine the impact of the change.

Before starting any benchmarks, temporarily disable `irqbalance` and `cpuspeed` on the Linux-based test client(s) to maximize network throughput and allow the best results:

Benchmark prep

```
service irqbalance stop
service cpuspeed stop
chkconfig irqbalance off
chkconfig cpuspeed off
```

Enable Jumbo Frames

Before proceeding, run a benchmark so the performance impact can be confirmed.

Increase the value of the maximum transmission unit (MTU) to enable jumbo frames. TCP uses the MTU to determine the maximum size of each packet in any transmission.

Important

Update this setting across all components of the Swarm implementation, including all switches and nodes.

Network switches

Replace "eth2" with the interface name to update the value to the interface config. See the switch manufacturer's instructions on how to change MTU size.

Caution

Verify the node's network interfaces and all network hardware support the selected MTU before the default value is changed; otherwise, the nodes may not be able to replicate objects or communicate.

<p>Swarm nodes</p>	<p>To update the value for the Swarm storage nodes, update the network.mtu setting in each node.cfg file and reboot. (This value is not stored in the persisted Settings object because it is specific to a given node.)</p> <p>Setting: network.mtu</p> <p>Default: 0 (use DHCP) or 1500</p> <p>Value: 9000</p> <p>Type: int</p> <p>Description: (Node-specific) in bytes. Sets the maximum transmission unit (MTU) Swarm accepts. Set to a higher value to use jumbo frames.</p>
<p>Other components</p>	<p>These include such components as test clients and Content Gateway. See the instructions for the component's operating system on how to change MTU size.</p>
<p>VMs</p>	<p>If the other components are VMs, follow VMware instructions on how to change MTU for ESXi hosts and guests.</p>

Change sysctl Settings

Before proceeding, run a benchmark so the performance impact can be confirmed.

Next, optimize the core memory settings in the Linux kernel.



RHEL 7

In RHEL 7, system tunables are set in the `/etc/sysctl.d/` directory, and they may be specified in more than one configuration file in this directory. The ordering logic determines which is used, so verify changes are not being overridden. See <https://access.redhat.com/solutions/800023>.

Gateway components and clients

Here is a modified `/etc/sysctl.conf`, which can be applied as sysctl changes. Swarm-specific recommendations are grouped at the end. Follow the operating system's recommendations and instructions for modifying sysctl settings.

Modified /etc/sysctl.conf

```
# -- tuning -- #
# Increase system file descriptor limit
fs.file-max = 65535

# Increase system IP port range to allow more concurrent connections
net.ipv4.ip_local_port_range = 1024 65000

# -- 10gbe tuning from Intel ixgb driver README -- #

# turn off selective ACK and timestamps
net.ipv4.tcp_sack = 0
net.ipv4.tcp_timestamps = 0

# memory allocation min/pressure/max.
# read buffer, write buffer, and buffer space
net.ipv4.tcp_rmem = 1000000 1000000 10000000
net.ipv4.tcp_wmem = 1000000 1000000 10000000
net.core.rmem_max = 524287
net.core.wmem_max = 524287
net.core.optmem_max = 524287

# Caringo-specific recommended values:
net.ipv4.tcp_mem = 134217728 134217728 134217728
net.core.rmem_default = 134217728
net.core.wmem_default = 134217728
net.core.netdev_max_backlog = 300000
```

These kernel updates are hardware/machine specific, so they are not saved to the cluster persistent settings.

Important

Swarm fails to start if an I/O error occurs while reading a sysctl setting.

Swarm storage nodes

There are per-chassis settings for network tuning available to set in the node.cfg file and can apply dynamically via SNMP. For sysctl-like and other buffer settings, it is possible to change them via SNMP, but these values are not stored in the [persisted Settings object](#) because they are specific to a given chassis.

Storage Setting	SNMP Name	Recommended	Type	Description
sysctl.deviceWeight	deviceWeight	256	int	Value of /proc/sys/net/core/dev_weight.
sysctl.tcpMem	tcpMem	134217728 134217728 134217728	str	Value of /proc/sys/net/ipv4/tcp_mem, in form 'min default max'.
sysctl.coreRMemDefault	rMemDefault	134217728	int	Value of /proc/sys/net/core/rmem_default.
sysctl.coreWMemDefault	wMemDefault	134217728	int	Value of /proc/sys/net/core/wmem_default.
sysctl.netdevMaxBacklog	netdevMaxBacklog	300000	int	Value of /proc/sys/net/core/netdev_max_backlog.
cip.readBufferSize		33554432	int	(Node-specific) In bytes. The size of the multicast UDP socket read buffer.

Set Buffer Size

Before proceeding, run a benchmark so the performance impact can be confirmed.

Swarm supports a limited number of tunable settings, such as buffer size.

Setting	Value	Type	Description
network.wmemMax	262144	int	Maximum value of wmem, ≥ 16384
network.rmemMax	262144	int	Maximum value of rmem, ≥ 87380
network.rxQueueLength	0	int	Value of ethtool -G ethX rx. 0 is unset, leaving kernel default

Proxying the Swarm Admin Console

Administrators running Swarm on a private, protected network may choose to allow clients on the external network to view the Swarm console without providing them access to the nodes themselves on the private network. Perform this by proxying the console from a privileged server that straddles the internal and external networks.

From a server running [Apache HTTP Server](#), the following rewrite rule for URLs can be applied using the [mod_rewrite](#) module:

```
<Location /storage/>  
  RewriteEngine On RewriteRule ./storage/([ /]+)/(.)$ http://$1:90/$2 [P,L]  
</Location>
```

Setting up the Swarm Network

Client applications must be able to initiate TCP connections with all nodes in a storage cluster using the designated access port, which is typically port 80. In a storage cluster, the nodes must be able to communicate with each other using UDP, TCP, and multicast communication protocols.

This section describes how to set up a storage cluster in a standard TCP/IP networking environment.

Network Communications

These are the required and optional network communications used in a storage cluster:

Communication	Protocol	Port	Client	Server	Requirement
Legacy Console	TCP	90	Administrator browser	Swarm storage node	Pulls information from a Swarm node into the legacy Swarm Admin Console
Swarm UI	TCP	91	Swarm Gateway	Swarm storage node, Gateway	Pulls information from a Swarm node into the Swarm Management UI. Gateway serves this to Administrator clients as service port termination.
SCSP	TCP	80	Client application (HTTP client)	Swarm storage node, Gateway	Required <div style="border: 1px solid gray; padding: 10px; margin-top: 10px;"> <p>Important</p> <p>Although unnamed objects are nearly impossible to guess by UUID, the metadata on those objects when cached in Elasticsearch (using the search feed mechanism) may reveal information about them through list and query operations. Secure port 80 from calls performing such operations.</p> </div>
SCSP	TCP	6000-6300	Swarm storage node	Swarm storage node	Required for SCSP communication between cluster nodes
Prometheus Node Exporter	HTTP	9100	Swarm Telemetry VM / Prometheus server	Swarm storage node, Gateway	Optional for Swarm telemetry
Health report	HTTPS	443	Swarm storage node	DataCore Health Report server	Required to publish Health Report information (see Health Data to Support)
DHCP	UDP	67	Swarm storage node	Swarm Cluster Services (SCS)	Recommended (Required when performing Platform network boot of Swarm storage nodes)
DNS	UDP	53	Swarm storage node	Swarm Cluster Services (SCS)	Optional
mDNS	UDP	5353	Swarm storage node	Swarm Cluster Services (SCS)	Optional
Multicast	UDP	7000	Swarm storage node	Swarm storage node	Required for Internode communication between cluster nodes
Internode messaging	UDP	9003-9300	Swarm storage node	Swarm storage node	Required for Internode communication between cluster nodes
NTP	UDP	123	Swarm storage node	Swarm Cluster Services (SCS)	Required
Logging	UDP	514	Swarm storage node	Swarm Cluster Services (SCS)	Required for Support
SNMP	UDP	161	SNMP client	Swarm storage node	Required for Support

TFTP	UDP	69	Swarm storage node	Swarm Cluster Services (SCS)	Required
Elasticsearch	TCP	9200	Swarm storage node, Gateway	Elasticsearch node	Required when using Gateway S3 and Content Portal
Elasticsearch Control Plane	TCP	9300	Elasticsearch node	Elasticsearch node	Required
Swarm License Server	TCP	8095	Swarm storage node	Swarm Cluster Services (SCS)	Required

IGMP Snooping

- [Disabling IGMP snooping](#)
 - [Warning](#)
- [Enabling an IGMP querier](#)
- [Node logging of IGMP snooping](#)

Disabling IGMP snooping

Managed switches may implement IGMP snooping to direct multicast traffic to ports. The purpose of IGMP snooping is to block unnecessary multicast traffic from hosts who are not interested in the traffic. The switch forwards multicast traffic out to ports where it heard an IGMP join message within a configurable time (typically around 5 minutes).

Swarm nodes are recommended to exist in a separate private VLAN so there are no other hosts in the broadcast domain. Disable IGMP snooping from the Swarm nodes' VLAN because there is no benefit to having IGMP snooping configured in a VLAN solely including Swarm products.

Configure an IGMP querier for the cluster's multicast group(s) if IGMP snooping from the VLAN cannot be disabled.



Warning

Enabling IGMP snooping without an IGMP querier results in nodes that cannot communicate with each other.

See the router documentation for details on enabling an IGMP querier.

For more information on IGMP Snooping, see [RFC 1112](#), [RFC 2236](#), and [RFC 3376](#).

Enabling an IGMP querier

When a Swarm node joins a cluster, it sends an initial set of unsolicited join requests for the configured multicast group. At that point, all Swarm nodes are visible from the Swarm Admin Console of all other nodes. IGMP queriers periodically send another query to see if there are any hosts still interested in the multicast group. As required by the IGMP RFCs, Swarm nodes do not send additional unsolicited join requests. If there is no querier for that multicast group in the network, the switch stops forwarding multicast traffic for that particular group out of that particular switch port when the switch timer for that multicast group runs out. After the timeout, all Swarm nodes appear to be unable to contact each other because the router did not send a query to prompt a subsequent join by the Swarm node.

Some switches are configured to act as IGMP queriers in an IPv4 network for multicast group memberships, but other switches are not unless configured appropriately. Since multicast routing is not configured by default on all switches, an IGMP querier may not exist on the network unless one has been configured.

Swarm can be configured to perform this function if a querier is not available in the network. Swarm elects a node in the cluster to act as an IGMP querier when configured, guaranteeing multicast membership queries are sent so cluster operation can continue. The Swarm IGMP querier has no impact on networks with IGMP Snooping disabled.

To enable an IGMP querier on the Swarm cluster itself, set it via the SNMP MIB entry `networkIGMPTimeout` and enable the [network.igmpTimeout](#) configuration parameter in Swarm.

Node logging of IGMP snooping

To help identify cluster networks where IGMP snooping is enabled *without* an IGMP querier, a node logs a critical error on the Swarm Admin Console and in the syslog, recommending administrators check for the presence of IGMP snooping. The node does this when these conditions occur:

- It previously multicasted to a node.
- It can no longer multicast to that node.
- It still unicasts to that node.

Swarm uses IGMPv2 responses to host membership queries by default. The `igmpVersion` parameter can be used to force the use of version 1, 2, or 3.

Hardware Setup

This section addresses how best to configure the hardware devices in your storage cluster.

- [Hardware Requirements for Elasticsearch Cluster](#)
- [Hot Swapping and Plugging Disks](#)
- [Multipath Support](#)
- [Hardware Requirements for Storage](#)
- [Hardware Booting](#)
- [Local Area Replication with Subclusters](#)

Hardware Requirements for Elasticsearch Cluster

- [Hardware Best Practices](#)
- [RAM for Elasticsearch](#)
- [Disk Usage for Search](#)
- [Optimizing Disk I/O for ES](#)
- [Optimizing Disaster Recovery for ES](#)

An Elasticsearch cluster supports Swarm searches. The Swarm feeds mechanism (see [Understanding Feeds](#)) populates the metadata search servers running the Elasticsearch (ES) software.

See [Elasticsearch Implementation](#).

i Info

Elasticsearch was previously used to store Historical Metrics but has moved to [Prometheus](#) starting with Swarm 14. Gateway [Content Metering](#) stores *csmeter indices* in Elasticsearch, but this does not impact Elasticsearch hardware requirements as much as a Swarm Search Feed.

This software requires one or more servers running RHEL/CentOS 7 Linux. Although Elasticsearch runs on other Linux platforms, DataCore currently provides support and testing for these versions. The Elasticsearch version provided with the Swarm distribution is supported.

See the [Elasticsearch project website](#) for more about Elasticsearch.

i Do not install on management node

Both the Content Gateway and the production Elasticsearch cluster need to be on separate machines from the management node (SCS). The management node installs with Service Proxy and a single-node ES, which are dedicated to the Swarm UI.

Hardware Best Practices

Following are overall best practices, with [hardware recommendations](#) from Elasticsearch:

- Provision the machines with CPUs with at least 4 cores and 64 GB memory. Between faster processors or more cores, choose more cores.
- Choose Solid-state drives (SSD), to boost performance. This is critical for S3, especially rapid small object writes, and for the listing of buckets with millions of objects.
- Perform the following if using hard disks which do not handle concurrent I/O as well as SSDs:
 - Select high-performance server disks.
 - Use RAID 0 with a writeback cache.
 - Set `index.merge.scheduler.max_thread_count` to 1, to prevent too many merges from running at once.

```
curl -X PUT <ES_SERVER>:9200/<SWARM_SEARCH_INDEX>/_settings \
-d '{ "index": { "merge.scheduler.max_thread_count": 1 } }'
```

- As with the storage cluster, choose similar, moderate configurations, for balanced resource usage.

RAM for Elasticsearch

RAM is key for Elasticsearch performance. Use these guidelines as a basis for capacity planning:

- 64 GB RAM per machine is optimal ([recommended by Elasticsearch](#)).
- Dedicate half of total RAM to the Java Virtual Machine (JVM) running Elasticsearch, but [do not exceed 31 GB](#), for best performance.
- Disable swapping of the Elasticsearch image. (For ES 2.3.3, allow in-memory caching of all shards on the server.)

Optimal performance can be achieved by adding adequate RAM in the ES servers to store all database shards in memory. Take steps to [disable or mitigate swapping](#). Memory page swapping on an ES server impacts Elasticsearch performance.

ⓘ Important

Watch for sustained increases in page swapping and disk I/O when monitoring the ES servers. This may mean additional RAM is needed on an ES server or additional servers need to be deployed to offset the load.

Disk Usage for Search

The storage on the Elasticsearch servers is used to persist the shards of the Swarm Search. Follow these guidelines for capacity planning for the Swarm Search indices.

- **Baseline metadata** to support listing: 150 GB per 200 million objects
- **Full metadata** to support ad-hoc searching: 300 GB per 200 million objects
- **Custom metadata**: allocate additional storage in proportion if indexing a large amount of custom metadata

These are unique *objects*, not *replicas*: how many Swarm replicas a Swarm object has is irrelevant to the ES servers. There is one metadata entry for the object no matter how many replicas of an object exist in the cluster.

ⓘ Tip

Do not confuse this with the RAM-based [Overlay Index](#) each *storage node* maintains, which depends on the total number of replicas in the cluster.

Optimizing Disk I/O for ES

Elasticsearch heavily utilizes disks, so higher throughput results in more stable nodes. Follow these [Elasticsearch guidelines](#) for optimizing disk I/O:

- **Use SSDs.** SSDs boost performance. With SSDs, verify the [OS I/O scheduler is configured correctly](#).
- **Use RAID 0.** Striped RAID increases disk I/O, at the expense of potential failure if a disk dies. Do not use mirrored or parity RAIDS, because replicas provide this functionality.
- **Do not use remote-mounted storage**, such as NFS or SMB/CIFS; the latency negatively impacts performance.
- **Avoid virtualized storage, such as a SAN or EBS** (Amazon Elastic Block Store). Even when SSD-backed it is often slower than local instance storage and it conflicts with the purpose of replicas and sharding.

Optimizing Disaster Recovery for ES

Elasticsearch clustering is designed to mitigate the impact of hardware and power failures, so long delays from refreshing the search data are not experienced. Determining how to invest and optimize hardware depends on how important metadata search and querying is to the organization and how long these features can be offline while Elasticsearch rebuilds data.

These are principles for making a configuration more disaster-proof:

- Do not use and rely on a single Elasticsearch server. This introduces vulnerabilities and potential disruption of search capabilities, and risks too little capacity to support all search needs.
- For power failure protection, deploy enough Elasticsearch servers to survive multiple server failures and distribute them across different power sources.
- Set up Elasticsearch with multiple nodes spread equally among the subclusters if the cluster is divided in to subclusters to match the power groups. This strategy improves survivability of a power group failure.

Hot Swapping and Plugging Disks

Administrators can insert a disk into a running node as long as the server hardware supports this function. Replacing failed disks ([hot swapping](#)) or adding additional disks (hot plugging) is supported without a server reboot.

Swarm recognizes, formats, and mounts the disk as a new volume when inserting a new unformatted disk. The disk continues to function as a volume without data loss when inserting a Swarm-formatted disk into the same node or different node. The volume remains retired if the formatted disk was previously retired.

No manual configuration or intervention is needed. Messages display in logs and in the Swarm Admin Console to indicate a disk was inserted or removed.

Requirements for Hot Plugging



Note

Not all hardware supports hot plugging in Swarm correctly. Contact an account representative to determine if hardware is supported.

- The configuration option **disk.volumes** must be set to **all**.
- JBOD/pass-through mode must be supported and enabled to use a disk with a RAID controller. Contact DataCore support for details.
- Disks must not be configured in RAID.
- Any virtual machines housing Swarm storage nodes must enable disk UUIDs (set [disk.EnableUUID=TRUE](#)).

Guidelines for Hot Swapping

- Count the total disks (status **OK**) across all node processes and verify it equals the value before pulling any disks to determine if hot swapping succeeded. The disk may not be assigned to the same node process handling it before being moved.
- Expect a disk that is plugged back in to least show up in the first node process on the machine if SNMP is slow to update. The disk add algorithm attempts to keep the volume assignments balanced across node processes.
- Check the disk identification lights: The disk identification light is automatically enabled if a disk cannot mount when hot-plugged into a system or fails at boot time.
- Expect to see "noise" in the syslog about failed volume recovery (FVR) starting when pulling a live, good disk (not retired or disabled due to error count). Look for these announcements:
 - FVR has completed or been cancelled on the hot-swapped disk.
 - The hot-swapped volumeID is mounted and recognized by the assigned node process.

```

10) Clear Announcements...
09:48 mounted /dev/sdbu(mechanical), volumeID is 7c1b43f98c80df0a8672f0e56888df58
11:18 FVR of remote volume 7c1b43f98c80df0a8672f0e56888df58 complete against 18 local vols
11:18 Finished ECR for remote volume 7c1b43f98c80df0a8672f0e56888df58, all done
11:18 Remote volume 7c1b43f98c80df0a8672f0e56888df58 is down; initiating FVR against 18 local volumes;
10:53 FVR of remote volume b09b5898638bc93807e3ae95c03a063a complete against 18 local vols
10:53 Finished ECR for remote volume b09b5898638bc93807e3ae95c03a063a, all done
    
```

Multipath Support

- [Understanding Multipath](#)
 - [Caution](#)
 - [How multipath assigns names](#)
 - [Tip](#)
 - [Caution](#)
 - [How multipath validates disks](#)
- [Configuring Multipath](#)
 - [Recommended settings](#)
 - [Important](#)
 - [When to disable multipath](#)
 - [Note](#)
- [Troubleshooting Multipath](#)
 - [Volumes are missing](#)
 - [Volumes cannot be physically located](#)
 - [Volumes are misnamed](#)
 - [Tip](#)

Understanding Multipath

Within Linux, *Device Mapper Multipathing* (DM-Multipath) provides I/O failover and load balancing for storage devices. Host servers use multiple paths of a redundant network to provide continuous availability and higher bandwidth connectivity with the devices. DM-Multipathing routes around path failures and balances the load across the paths.

- **The problem** – Higher-end hardware is offering DM-Multipath support, which provides failover protection through data path redundancy (via multiple cables). Without multipath support, Swarm counts the disks in such a chassis twice because it sees them over both cables (if a chassis has 60 disks, Swarm sees 120 volumes to manage).
- **The solution** – With multipath support, Swarm can handle redundant cables as one path. This handling prevents Swarm from adding the same physical disk through different volume names. Swarm monitors and updates the multipath mappings for all hot-plug events, such as adding a physical disk, a cable on the multipath device, or a new multipath chassis. With multipath, Swarm makes dynamic accommodation when redundant cables fail or are hot plug added or removed:
 - No Swarm volumes are affected when a *second* cable is gained or lost on a multipath chassis.
 - All Swarm volumes on the chassis are affected when the *sole* cable is gained or lost.

Caution

Changing the multipath configuration setting requires an immediate cluster reboot and renames all disks.

How multipath assigns names

The most visible effect of multipath support is how it maps volumes to new names:

- **Enabled:** With multipath enabled, Swarm assigns *every* disk in a cluster a multipath name in a pattern like `/dev/dm-x`, even if none of the existing disks support multipath. Swarm manages the mapping between the multipath name and the underlying disk name.

Tip

Multipath is not enabled in a cluster if Swarm shows names without the prefix `dm-` (which refers to DM-Multipath).

- **Disabled:** *Every* disk in a cluster keeps the volume name in a pattern like `/dev/sdx`, even if it is a multipath disk with multipath disabled.

Caution

Without multipath, Swarm cannot manage redundant connections.

How multipath validates disks

Swarm has validations to prevent physical disks from being mounted more than once and to keep disk controller errors from inconsistently mapping multipaths:

- During startup and hot-plug events, Swarm checks disk volume headers to verify a physical disk is not mounted through multiple paths and it is referred to through a multipath volume name.

- Swarm compares multipath mappings through different command sequences to verify they each provide the same data.
- Swarm partitions disks in to sets with the same volume header and compares them to multipath mappings to verify mappings and partitions are mutually consistent.

Configuring Multipath

You control multipath by configuring a pair of options in the settings file:

- `disk.enableMultipath`
- `disk.volumes`

The `disk.enableMultipath` setting accepts integers, with 0 meaning false and all other values meaning true.

Recommended settings

Multipath is disabled by default, so careful planning is possible when enabling it and rebooting the cluster:

```
disk.enableMultipath = 1 # Enable multipath; disabled by default
disk.volumes = all      # Multipath vols assigned to nodes by Swarm
```

The best practice is to disable multipath if named volumes (with or without `except`) need to be specified:

```
disk.enableMultipath = 0 # Disable multipath
disk.volumes = /dev/sda /dev/sdb
```



Important

Hot plugging is supported for `disk.volumes = all`.

When to disable multipath

Do not disable multipath on the assumption this boosts performance. Any performance differences are negligible.

For best results, review the situation with Support and enable multipath unless Support advises against it. Having multipath enabled allows Swarm to handle the disks in a multipath, multi-channel chassis correctly and seamlessly, if any are added later.

Support may advise disabling multipath (`disk.enableMultipath=0`) in a few cases:

- Not using multipath and do not intend to ever use such hardware.
- **Disk.volumes = all** is not set.



Note

Disable multipath and use a single channel of the multipath chassis if assigning certain disks to Swarm. It is challenging to predict the resulting volume names for each disk.

Troubleshooting Multipath

Volumes are missing

A disk controller problem may be the cause if there are missing volumes (or no volumes) mounted from a multipath chassis. Look for a disk controller in the multipath chassis behaving incorrectly.

Volumes cannot be physically located

Check the configuration settings and verify multipath is not enabled while specific volumes are explicitly assigned, which causes confusion: There is no way to know which multipath volumes correspond to particular physical disks because Swarm assigns multipath volume names non-deterministically.

Contact DataCore Support for help finding the multipath device name for a physical disk.

Volumes are misnamed

All volume names (including non-multipath volumes) *must* appear with multipath naming, such as `/dev/dm-1`, `/dev/dm-2` unless multipath is disabled. This may be alarming if non-multipath volumes are expected to retain non-multipath naming, such as `/dev/sda`.



Tip

Contact DataCore Support for help adjusting processes so they do not require physical volume names.

Hardware Requirements for Storage

This section describes the hardware requirements for implementing a storage cluster in a corporate enterprise and the best practices for maintaining it.

Note

Swarm installs and runs on enterprise-class (not consumer-grade) x86 commodity hardware.

Caution

Configure a cluster with a minimum of three nodes to guarantee high availability and fail over in the event of a node failure.

- [Virtualization](#)
- [Minimum Requirements](#)
- [Recommended Requirements](#)
- [Memory Sizing Requirements](#)
- [Supporting Erasure Coding](#)
- [Supporting High-Performance Clusters](#)
- [Balancing Resources](#)
- [Selecting Hard Disks](#)
- [Mixing Hardware](#)

Virtualization

Swarm storage nodes can run in a VM environment. Swarm supports VMware/ESXi and Linux KVM. Contact sales for more information and guidance.



Best practice

Enable volume serial numbers on any virtual machines housing Swarm storage nodes (set [disk.EnableUUID=TRUE](#)).

Minimum Requirements

The following table lists the minimum hardware requirements for a storage cluster. Because Swarm nodes are designed to run using lights-out management (or out-of-band management), they do not require a keyboard, monitor, and mouse (KVM) to operate.

Component	Requirement
Node	x86 with Pentium-class CPUs
Number of nodes	Three (to guarantee adequate recovery space in the event of node failure)
Switch	Nodes must connect to switches configured for multicasting
Log server	To accept incoming syslog messages
Node boot capability	USB flash drive or PXE boot
Network interfaces	One Gigabit Ethernet NIC with one RJ-45 port *
Hard disks	One hard disk
RAM	2 GB + 0.5 GB × number of volumes (light loads and no erasure-coding)
NTP server	To synchronize clocks across nodes

* All nodes in a cluster must use the same speed network interface to be compatible.

Recommended Requirements

The following table lists the recommended hardware requirements for a storage cluster.

Component	Requirement
Node	x86 with Intel Xeon or AMD Athlon64 (or equivalent) CPUs
Number of nodes	Three or more
Switch	Nodes must connect to switches configured for multicasting
Log server	To accept incoming syslog messages
Node boot capability	USB flash drive or PXE boot
Network interfaces	Two Dual Gigabit Ethernet NICs with two RJ-45 ports for link aggregation (NIC teaming) Important: <i>Mixing network speeds among nodes is not supported.</i> Do not put a node with a 100 Mbps NIC in the same cluster with a node containing a 1000 Mbps NIC.
Hard disks	One to four standard non-RAID SATA hard disks
RAM	4 GB + 0.5 GB × number of volumes
NTP server	To synchronize clocks across nodes

Although Swarm runs on a variety of x86 hardware, this table lists the recommended base characteristics for a storage cluster. Cluster performance improves exponentially by adding additional systems with more intensive CPU hardware and additional memory.

What hardware is best depends on storage and performance requirements, so ask a sales representative for hardware recommendations specific to the needs of the business.

Memory Sizing Requirements

Review the following sections for factors influencing how memory and erasure coding are sized, as well as how to configure Swarm.

How RAM affects storage

The storage cluster is capable of holding the sum of the maximum object counts from all nodes in the cluster. How many objects can be stored on a node depends both on the node's disk capacity and the amount of system RAM.

The following table shows estimates of the maximum possible number of replicated objects (regardless of size) that can be stored on a node, based on the amount of RAM in the node, with the default 2 replicas being stored. Each replica takes one slot in the in-memory index maintained on the node.

Amount of RAM	Max. immutable objects	Max. alias or named objects
4 GB	33 million	16 million
8 GB	66 million	33 million
12 GB	132 million	66 million

How the Overlay Index affects RAM

Larger clusters (those above 3 nodes by default) need additional RAM resources to take advantage of the Overlay Index.

To store the same number of reps=2 object counts above and utilize the Overlay Index, increase RAM as follows:

- Immutable unnamed objects: **50%** additional RAM
- Alias or named objects: **25%** additional RAM

Smaller clusters and larger clusters where the Overlay Index is disabled do not need this additional RAM.

See [Configuring the Overlay Index](#).

How to configure for small objects

Swarm allows storage of objects up to a maximum of 4 TB. Configure the storage cluster accordingly if storing mostly *small* files.

Swarm allocates a small amount of disk space to store, write, and delete the disk's file change logs (journals) by default. This default amount is plenty in typical deployments because the remainder of the disk is filled by objects before the log space is consumed.

The file log space can fill up before the disk space for installations writing mostly small objects (1 MB and under). Increase the configurable amount of log space allocated on the disk before booting Swarm on the node for the first time if a cluster usage focuses on small objects.

The parameters used to change this allocation differ depending on the software version in use.



Tip

Contact DataCore Support for guidance on setting the optimal parameters for the configuration.

Supporting Erasure Coding

Erasure coding conserves disk space but involves additional calculations and communications. Anticipate an impact on memory and CPU utilization.

Takeaway

Erasure coding uses about half the space of replication but it requires more RAM and other resources.

CPU	In general, plan to support EC with more and faster CPU cores.
Memory	<p>Scale up for larger objects: Larger objects require more memory to manage erasure sets. How many EC objects can be stored on a node per GB of RAM depends on the size of the object and the encoding specified in the configuration. The erasure-coding manifest takes two index slots per object, regardless of the type of object (named, immutable, or alias). Each erasure-coded segment in an erasure set takes one index slot. Larger objects can have multiple erasure sets, so multiple sets of segments exist. In k:p encoding (integers for the <i>data</i> (k) and <i>parity</i> (p) segment counts), there are p+1 manifests (up to the ec.maxManifests maximum). That means 3 manifests for 5:2 encoding. With the default segment size of 200 MB and a configured encoding of 5:2:</p> <ul style="list-style-type: none"> • 1-GB object: (5+2) slots for segments and (2+1) for manifests = 10 index slots • 3-GB object: 3 sets of segments @ 10 slots each = 30 index slots <p>Increase for Overlay Index: Larger clusters (above 3 nodes by default) need additional RAM resources to take advantage of the Overlay Index. For erasure-coded objects, allocate 10% additional RAM to enable the Overlay Index.</p>
Network	Network use by the requesting SAN is an important factor in EC performance. The requesting SAN must orchestrate k+p segment writes (for an EC write) or k segment reads (for an EC read). Because these segment reads and writes must occur at the same rate, <i>the slowest one slows the overall request</i> . Consequently, when a cluster experiences a lot of EC requests, nodes can play different roles and slower nodes can affect multiple requests.

See [Erasure Coding EC](#).

Supporting High-Performance Clusters

Swarm benefits from faster CPUs and processor technologies, such as large caches, 64-bit computing, and fast Front Side Bus (FSB) architectures for the demands of high-performance clusters.

Maximize these variables to design a storage cluster for peak performance:

- Add nodes to increase cluster throughput – like adding lanes to a highway
- Fast or 64-bit CPU with large L1 and L2 caches

Important

Disable this feature within the BIOS setup to prevent single-CPU degradation in Swarm if the cluster node CPU supports hyper-threading.

- Fast RAM BUS (front-side BUS) configuration
- Multiple, independent, fast disk channels
- Hard disks with large, on-board buffer caches and Native Command Queuing (NCQ) capability
- Gigabit (or faster) network topology between all storage cluster nodes
- Gigabit (or faster) network topology between the client nodes and the storage cluster

Balancing Resources

Attempt to balance resources across nodes as evenly as possible for best performance. Adding several new nodes with 70 GB of RAM can overwhelm those nodes and negatively impact a cluster of nodes with 7 GB of RAM.

Creating a large cluster and spreading the user request load across multiple storage nodes significantly improves data throughput because Swarm is highly scalable. This improvement increases as nodes are added to the cluster.

Tip

Using multiple replicas when storing objects in the cluster is an excellent way to get the most out of Swarm, because each copy provides redundancy and improves performance.

Selecting Hard Disks

Selecting the right hard disks for the storage nodes improves both performance and recovery, in the event of a node or disk failure. Follow the guidelines below when selecting disks. For an in-depth overview of disk characteristics, download the Intel white paper [Enterprise class versus Desktop class Hard Drives](#).



Best practice

Consult with DataCore Support prior to purchasing additional hardware to guarantee maximum compatibility.

Disk type	Enterprise-level	<p>The critical factor is whether the hard disk is designed for the demands of a cluster. Enterprise-level hard disks are rated for 24x7 continuous-duty cycles and have time-constrained error recovery logic suitable for server deployments where error recovery is handled at a higher level than the on-board controller.</p> <p>In contrast, consumer-level hard disks are rated for desktop use only; they have limited-duty cycles and incorporate error recovery logic that can pause all I/O operations for minutes at a time. These extended error recovery periods and non-continuous duty cycles are not suitable or supported for Swarm deployments.</p>
Reliability	Rated for continuous use	<p>The reliability of hard disks from the <i>same</i> manufacturer vary, because the disk models target different intended use and duty cycles:</p> <ul style="list-style-type: none"> • Consumer models targeted for the home user and assume the disk is not being used continuously. These disks do not include the more advanced vibration and misalignment detection and handling features. • Enterprise models targeted for server applications and tend to be rated for continuous use (24x7) and include predictable error recovery times, as well as more sophisticated vibration compensation and misalignment detection.

<p>Performance</p>	<p>Large on-board cache</p> <p>Independent channels</p> <p>Fast bus</p>	<p>Optimize the performance and data throughput of the storage sub-system in a node by selecting disks with these characteristics:</p> <ul style="list-style-type: none"> • Large buffer cache – Larger, on-board caches improve disk performance. • Independent disk controller channels – Reduces storage bus contention. • High disk RPM – Faster-spinning disks improve performance. • Fast storage bus speed – Faster data transfer rates between storage components, a feature incorporated in these types: <ul style="list-style-type: none"> • SATA-300 • Serial Attached SCSI (SAS) • Fibre Channel hard disks <p>Use of independent disk controllers is often driven by the storage bus type in the computer system and hard disks.</p> <ul style="list-style-type: none"> • PATA – Older ATA-100 and ATA-133 (or Parallel Advanced Technology Attachment [PATA]) storage buses allow two devices on the same controller/cable. Bus contention occurs when both devices are in active use. Motherboards with PATA buses typically only have two controllers. Some bus sharing must occur if more than two disks are used. • SATA – Unlike PATA controllers, Serial ATA (SATA) controllers and disks include only one device on each bus to overcome the previous bus contention problems. Motherboards with SATA controllers typically have four or more controllers. Recent improvements in Serial ATA controllers and hard disks (commonly called SATA-300) have doubled the bus speed of the original SATA devices.
<p>Recovery</p>	<p>Avoid highest capacity</p>	<p>Improve the failure and recovery characteristics of a node when a disk fails by selecting disks with server-class features but not the highest capacity.</p> <ul style="list-style-type: none"> • Higher capacity means slower replication. Consider the trade-off between the benefits of high-capacity disks versus the time required to replicate the contents of a failed disk when choosing the disk capacity in a node. Larger disks take longer to replicate than smaller ones, and that delay increases the business exposure when a disk fails. • Delayed errors mean erroneous recovery. Unlike consumer-oriented devices, for which it is acceptable for a disk to spend several minutes attempting to retry and recover from a read/write failure, redundant storage designs such as Swarm need the device to emit an error quickly so the operating system can initiate recovery. The entire node may appear to be down, causing the cluster to initiate recovery actions for all disks in the node – not the failed disk if the disk in a node requires a long delay before returning an error. • Short command timeouts mean less impact. The short command timeout value inherent in most enterprise-class disks allows recovery efforts to occur while other system disks continue to support system disk access requests by Swarm.

<p>Controllers and RAID</p>	<p>JBOD, not RAID Controller-compatible</p>	<p><i>Best practice:</i> Check with DataCore before investing in new equipment, both for initial deployment and for future expansion of a cluster. DataCore can help avoid problems not only with disk controller options but also with network card choices.</p> <ul style="list-style-type: none"> • Evaluate controller compatibility before each purchasing decision. • Buy controller-compatible hardware. The more types of controllers in a cluster, the more restrictions exist on how volumes can be moved. Study these restrictions, and keep this information with the hardware inventory. • Avoid RAID controllers. Always choose JBOD over RAID when specifying hardware for use in Swarm. RAID controllers are problematic, for these reasons: <ul style="list-style-type: none"> • Incompatibilities in RAID volume formatting • Inability of many to hot plug, so the ability to move volumes between machines is lost • Problems with volume identification (disk lights) • HP Proliant systems with P840 RAID controllers - Swarm performance degrades when this card runs in HBA mode. For best performance, use only single raid 0s per disk, but volumes in this system cannot hotplug.
<p>Firmware</p>	<p>Track kernel mappings</p>	<p>Keep track of controller driver to controller firmware mappings in the kernel shipped with Swarm Storage. This is particularly important when working with LSI-based controllers (which are the majority), because a mismatch between driver and firmware in LSI's Fusion MPT architecture can introduce indeterminate volume behavior (such as good disks reporting errors erroneously due to a driver mismatch).</p>

Mixing Hardware

Swarm greatly simplifies hardware maintenance by making disks independent of a chassis and disk slots. As long as disk controllers are compatible, move disks as needed. Swarm supports a variety of hardware, and clusters can blend hardware as older equipment fails or is decommissioned and replaced. The largest issue with mixing hardware is incompatibility among the disk controllers.

Follow these guidelines for best results with mixing hardware:

<p>Track controllers</p>	<p>Monitor the hardware inventory with special attention to the disk controllers when administering the cluster. Some RAID controllers reserve part of the disk for controller-specific information (DDF). Once a volume is formatted for use by Swarm, it must be used with a chassis having that specific controller and controller configuration.</p> <p>Many maintenance tasks involve physically relocating volumes between chassis to save time and data movement. Use the inventory of the disk controller types to spot when movement of formatted volumes is prohibited due to disk controller incompatibility.</p>
<p>Test compatibility</p>	<p>To determine controller compatibility safely, test <i>outside</i> of the production cluster.</p> <ol style="list-style-type: none"> 1. Set up two spare chassis, each with the controller being compared. 2. Format a new volume in Swarm in the first chassis. 3. Move the volume to the second chassis and watch the log for error messages during mount or for any attempt to reformat the volume. 4. Retire the volume in the second chassis and move it back to the first. 5. Watch for errors or attempts to reformat the volume. 6. Erase the disk using <code>dd</code> and repeat the procedure in reverse, where the volume is formatted on the second chassis if all goes well. <p>Swap volumes between these chassis within a cluster if no problems occur during this test. Do not swap volumes between these controllers if this test runs in to trouble.</p>

Hardware Booting

The Swarm node hardware needs to be configured to boot either from the USB flash drive or a [PXE](#) network (which is the default method for the Platform Server).

See [Setting up PXE Booting](#).

Platform Server

Skip this section if using [Platform Server](#): the hardware is set up.

Booting from USB

The hardware may identify the device as either a hard disk or a removable device if the cluster boots from a USB device. Verify the USB device appears at the top of the boot priority order so the system boots from the USB drive before any other devices on the node.

Because all internal disks are typically used for storage, the BIOS may prevent booting from a hard disk. Verify the boot priority of the hard disks is lower than the USB flash drive.

Keyboard Errors

A keyboard attached to the cluster node is not required during normal Swarm operations.

Note

The cluster node BIOS may need to be configured to avoid keyboard errors during boot.

Local Area Replication with Subclusters

Local area replication (LAR) allows logical separations to be created in a storage cluster to define storage distribution strategies. These logical separations cause Swarm to attempt to create the greatest logical spread between an object's replicas by moving them into separate *subclusters*. Examples where LAR subclusters are useful include:

- Splitting a cluster based on **location** (data cabinet, building wing)
- Grouping nodes based on **common infrastructure** (network, power)

Consider splitting a cluster based on location if data centers are located in separate wings of a building and copies of stored content is desired to exist in both locations in case of a partial building loss. Grouping the cluster nodes based on a common infrastructure can be another option, if wanting to group cluster nodes by shared network switches, a common power distribution unit (PDU), or a common electrical circuit within a rack.



Note

Multi-server subclusters are a special case. Even though each node is assigned a chassis-level default subcluster, it can be overridden to configure a different subcluster.

Network requirements. The network connections between LAR subclusters must have the same speed and latency characteristics as the connections between the nodes. All nodes must be in the same broadcast domain so they are able to send data directly to all nodes in the cluster and receive the multicast traffic sent from anywhere in the cluster.



Warning

Minimize moving nodes to new subcluster assignments, which can cause high cluster activity while content is redistributed to conform to the new subcluster assignments.

Space requirements. Verify sufficient space exists in the LAR subcluster containing the retiring volumes if the separation to persist when retiring a volume is desired. Retiring a volume without sufficient space can be problematic because Swarm *must* maintain the correct number of replicas in the subcluster. Swarm may create all replicas on the other side of the subcluster, filling up that side of the subcluster.

See *node.subcluster* in the [Settings Reference](#).

Storage Implementation

This section covers Swarm infrastructure and implementation:

- **Network:** Set up the Swarm network: it provides examples of a network configuration and describes how to set up your Swarm network infrastructure, including the network services, system booting, and the network devices.
- **Hardware:** Select system and storage hardware, with requirements and recommendations on how to set up the hardware devices in your storage cluster
- **Installation:** Install and configure a basic storage cluster, including information about licensing, booting your Swarm nodes, using the System Menu, and using Syslog.

See also these sections:

- [Swarm Storage Release Notes](#)
- [Swarm Storage Cluster](#) (configuration and administration)
- [Storage SCSP Development](#)



Tip

Take advantage of the technical training videos available.

[Training video topics](#) cover Swarm components, administration, management, APIs, and user interfaces.

- [Installing and Initializing Swarm Storage](#)
- [Swarm Passwords](#)
- [Licensing Swarm](#)
- [Implementation Checklist](#)
- [Drive Identification Plugin](#)
- [Configuring Swarm for Gateway](#)

Installing and Initializing Swarm Storage

This section describes how to set up and configure a storage cluster. If you are upgrading an existing storage cluster, see [How to Upgrade Swarm](#).

If you are installing Swarm for the first time, your general path depends on the configuration you are using:

CSN with Gateway	CSN, no Gateway	No CSN
<ol style="list-style-type: none"> 1. Download CSN/Swarm from the Downloads section on the DataCore Support Portal. 2. Run the CSN bundle script. 3. Boot the cluster. 4. Install Elasticsearch. 5. Install Gateway (with Swarm UI and Content UI). 	<ol style="list-style-type: none"> 1. Download CSN/Swarm from the Downloads section on the DataCore Support Portal. 2. Run the CSN bundle script. 3. Boot the cluster. 4. To have a local Swarm UI on the CSN, run the Service Proxy script. 	<ol style="list-style-type: none"> 1. Download the Swarm bundle from the Downloads section on the DataCore Support Portal 2. Install Storage (fsimage/kernel files via USB key or PXE server). 3. Boot the cluster. 4. If using, install Elasticsearch. 5. If using, install Gateway (with Swarm UI and Content UI).

- [Storage Settings Checker](#)
- [Upgrading a Storage Cluster](#)
- [Using the System Menu](#)
- [Rebooting the Storage Cluster](#)
- [Initializing a Storage Cluster](#)

Storage Settings Checker

- [Running the Checker \(CSN\)](#)
- [Running the Checker \(Non-CSN\)](#)

Swarm Storage versions 10 and higher are supported by a configuration checker, which is bundled with the support tools downloaded from the Support site. The report establishes the current configuration across the cluster and surfaces any setting issues (such as deprecations and new requirements) that need to be addressed. (v10.0)

To evaluate all active settings for the Swarm cluster, the checker requires both types of configuration settings:

- **Dynamic** – The persisted settings stream (PSS object) for the cluster, which is updated in real time by the Swarm UI or SNMP.
- **Static** – The configuration file(s) used at node startup.

Best practice

Before troubleshooting Storage or starting an upgrade (versus a point update, such as from 11.0.1 to 11.0.3), download and run the latest settings checker and verify the results with Support.

Running the Checker (CSN)

For CSN sites only, use the `techsupport-bundle-grab.sh` with the `-s` option to run the settings checker and bundle the output along with the other needed logs and support information:

1. Download the latest support bundle: [swarm-support-tools.tgz](#) on the [Support site](#)

Important

Always verify the latest downloaded version is available when using this tool for settings analysis before upgrading.

2. Unpack the support tools.
3. Generate the complete tarball by running the following script with the settings:

```
techsupport-bundle-grab.sh -s
```

Note: The admin password is required if changed from the default of `admin:caringo`.

```
techsupport-bundle-grab.sh -s -A password_of_cluster_admin
```

4. Create a new ticket for the configuration review on the [Support site](#) and note the ticket number (such as SUP-1234), which is required later.
5. Upload the tarball.

- Run the following command from the directory where the support tools are located if the machine has access to the internet:

```
source bashrcforcustomers
```

Then run:

```
uploadtosupport [output-file]
```

- Use `scp` to secure copy the file to a location with internet access and then use the [Support Uploader](#) if no internet access is available.

6. Enter the ticket number when prompted; the tarball is attached to the ticket .
7. Support is notified of the upload and works on any configuration issues that are surfaced.

Running the Checker (Non-CSN)

For non-CSN sites run the settings checker as a standalone Python 3 script.

i Python 3 not available

Follow this process if Python 3 is not available:

1. On a RHEL/CentOS 6/7 server, download and unpack the support tools: [swarm-support-tools.tgz](#)
2. Run the following and collect the outputs:


```
platform-read-pss.sh -A [admin pass] -a [Swarm node IP] -S [snmp r/w pass]
hwinfo-dmesg-grab.sh -m [Swarm node IP]
```
3. Collect all node configuration files. Several may exist if a custom PXE server is used.
4. Zip together all configurations and script outputs above and upload to the [Support Uploader](#) as described below.

1. Download the latest support bundle: [swarm-support-tools.tgz](#) on the [Support site](#)
2. Check whether Python 3 is installed:

```
python --version
```

[Install it](#) now if it is not.

3. Unpack the support tools.
4. Run the following Python script with the needed options:

```
python3 settings_checker.py [options]
```

See **Script Options** below.

5. The script outputs a text file that does not need to be compressed. Use `scp` to secure copy the file to a location with internet access.
6. Create a new ticket for the configuration review on the [Support site](#) and note the ticket number (such as SUP-1234).
7. Upload the file with the [Support Uploader](#).
8. Enter the ticket number when prompted; the file is attached to the ticket.
9. Support is notified of the upload and works on any configuration issues

Script Options

Which options are required depends on what type of Swarm environment is running:

- CSN environments - None of these options are required if the `/var/opt/caringo/netboot/content/cluster.cfg` file has the correct admin and snmp r/w passwords.
- Non-CSN environments - Options (user, settings) are required to enable locating and loading of the dynamic settings (PSS).

i Typical usage, non-CSN

- *Locate the settings config file* – Add `f FILE` (for one or more file paths).
- *Locate the PSS (persisted settings stream)* – Add `-a [storage node ip]` and `-w [admin password]`.
- Add `c` to see the output on the console.

Options	Rules	Scope	Notes
-h, --help			Show program help and exit.
-v, --version			Show program's version number and exit.
-o OUTPUT			The pathname of the output file, which opens in append mode. Defaults to timestamped output in the current working directory: <code>./swarm_settings_\${HOSTNAME}_v\${TOOLVERSION}_\${TIMESTAMP}.out</code>
-c, --console			Disabled by default. Print to the console, as well as the output file.
-m			Display more output.
-w PASSWORD	<i>Do not use with -u</i>	CSN	The password of the cluster admin, used to retrieve the cluster's PSS (persisted settings stream). The password is read from the cluster.cfg file if not specified.
-u USERNAME_PASSWORD	<i>Do not use with -w</i>	Non-CSN	The <code>username:password</code> of the cluster admin, which is needed to read the PSS. The username is read from the node.cfg file if not specified.
-p PATH, --path=PATH	<i>Do not use with -f</i>	Non-CSN	The script defaults to legacy CSN configuration locations if not specified.
-g SUFFIX	<i>Only use with -p</i>	Non-CSN	Optionally, specifies the file type, if other than <code>.cfg</code> .
-f FILE [FILE ...], --file FILE [FILE ...]	<i>Do not use with -p</i>	Non-CSN	The location of the configuration file (or files separated by a space), or the path to all configuration files. Defaults to file type <code>.cfg</code>
-s SETTINGS	<i>use -a if not known</i>	Non-CSN	The host/UUID of the PSS for the cluster if known, in URI form: <code>http://HOST/UUID</code> .
-a ADDRESS	<i>Only provide if requested</i>	Non-CSN	The IP of a node in the cluster, to help obtain the PSS.
-r PSSFILE		Non-CSN	The location of a PSS file.

Upgrading a Storage Cluster

This section details how to upgrade a Swarm license and cluster nodes for non-CSN clusters.

For release-specific guidance, see [How to Upgrade Swarm](#) and [CSN Upgrades](#).

- [Types of upgrades](#)
 - [Errors](#)
- [Preparing for the upgrade](#)
 - [Important](#)
- [Upgrading the nodes](#)
 - [Tip](#)
- [Example shutdown script](#)

Types of upgrades

A single cluster can contain nodes running mixed versions during the upgrade process and no data conversion between versions is necessary unless noted for a release.

- **Simple upgrade** – The simplest upgrade method is to reboot the entire cluster at once after the software on all USB flash drives or the centralized configuration location has been updated.
 1. Shut down all nodes in the cluster.
 2. Upgrade the software.
 3. Reboot the nodes.
 4. Verify all nodes are healthy.
- **Rolling upgrade** – Restart the nodes one at a time with the new version and allow the cluster to continue serving applications during the upgrade process to upgrade the cluster without scheduling an outage or bringing down the cluster. Objects continue to be fully accessible during the upgrade if stored with at least two replicas. Wait at least 10 seconds between each node reboot to verify each node can properly communicate the rebooting state to the rest of the cluster and verify other cluster nodes do not initiate recovery for the rebooting node if using the rolling upgrade approach.



Errors

- **Ongoing processes** – Errors similar to 'Castor-System-Cluster value must refer to a remote cluster on RETRIEVE request' may be logged if there are any disk recovery or retire processes ongoing in the cluster during a rolling upgrade. These errors are harmless and stop once all nodes are running the new version.
- **Blocked feeds** – Swarm 9 modifies the feed definition in the persisted Settings object when starting a rolling upgrade from Swarm 8. The change is not supported by Swarm 8, so those nodes get blocked feeds with a config error ("Plugin validation error: Unknown attribute indexAlias") and are unblocked when the last Swarm node has been upgraded. During the rolling upgrade, the feed is blocked on some nodes, which may not support indexing and querying. The feed blocks again in the same way if ever downgrading to Swarm 8: either delete the feed and redefine it, or contact DataCore Support for help updating the feed definition in the persisted Settings object.

Preparing for the upgrade

To prepare for the upgrade:

1. Download the upgraded Swarm software from the [Downloads section](#) on the [DataCore Support Portal](#).
2. **Important:** Review the [release notes](#) for upgrade instructions specific to the version downloaded.
3. **Important:** Run the [Storage Settings Checker](#) and resolve any configuration issues with DataCore Support.
4. Locate the node configuration data, backup configuration files, and license files.
5. Prepare the node configuration data on new USB flash drives or on a centralized configuration server.
6. Verify the health of all cluster nodes.
7. Schedule an off-line window for the cluster down time.

Review the release notes included with the new boot devices prior to starting the cluster upgrade. The release notes contain information about feature changes, operational differences, and any issues affecting how a storage cluster processes and stores data.

Remove USB flash drives from the running nodes to view and back up the configuration and license files. USB flash drives or the configuration server can be updated using the instructions in the `README.txt` file found in the latest Swarm installation package. Validate the node or cluster configuration file to verify there are no deprecated parameters needing to be removed or [renamed](#) after performing the upgrade.

See [Configuring the Nodes](#) for how to set the parameters in the node or cluster configuration files.

Return each USB flash drive to the node from which it was removed after all upgrades and validations are completed. Match each USB device to the original node in the cluster and verify the `vols` parameter which defines the storage devices matches the correct node.

Important

Verify the cluster health by checking for critical error messages on the status page of each node or the SNMP `CastorErrTable` OID before performing any node upgrade. This process verifies no hardware problems exist that can interrupt the upgrade process. Any problems need to be corrected prior to upgrading the cluster.

When upgrading a single node in a cluster, include the `clusterSettingsUUID` parameter value in the node or cluster configuration file prior to rebooting the node so the settings file can be located after the nodes reboot.

Upgrading the nodes

To upgrade the cluster nodes:

1. Shut down all cluster nodes (or one at a time for a rolling upgrade).
2. Install the updated USB flash drives on the PXE boot server.
3. Reboot all nodes.
4. Verify the cluster is operating normally.

A simultaneous shutdown of all cluster nodes is the first step in the simple upgrade. The nodes can be rebooted one at a time in a rolling upgrade so the cluster remains online if the cluster cannot be shut down during normal business operations, .

The cluster detects the missing node and the remaining nodes attempt to recover the missing content via the failed volume recovery (FVR) and erasure coding recovery (ECR) processes if performing a rolling upgrade and a cluster node is off-line. The cluster detects the node and the recovery processes stop when the missing node is brought back online.

Tip

Prepare USB flash drives with the upgraded version of Swarm for each node before beginning the upgrade to minimize node downtime and prevent the remaining nodes from filling with recovered content.

Initiating the disk recovery process is not a concern if all nodes are shut down within several seconds of each other. Suspending volume recovery may also be chosen from the **Setting** window on the Swarm Admin Console to prevent recovery from kicking off while nodes reboot in to the new software version.

See the [Suspend Setting](#).

Example shutdown script

This UNIX shell script demonstrates a method of issuing the shutdown command to all cluster nodes. In this example, all nodes in the cluster are defined in the NODES variable.

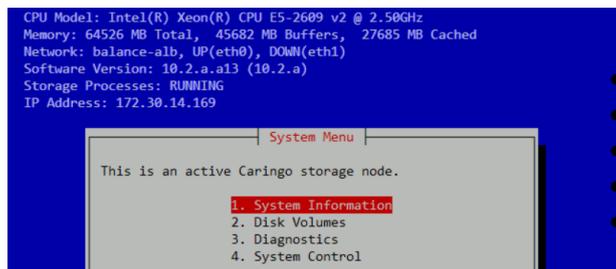
```
NODES="192.168.1.101 192.168.1.102 192.168.1.103"
for n in $NODES;
do snmpset -v 2c -c pwd -m +CARINGO-CASTOR-MIB
  $n caringo.castor.CastorShutdownAction = "shutdown"
done
```

Using the System Menu

- [About the System Menu](#)
 - [Disabling for Security](#)
- [Menu Options](#)
 - [Caution](#)
- [Formatting a Disk](#)
 - [Note](#)
- [Reformatting all Disks in a Cluster](#)
 - [Note](#)

About the System Menu

The System Menu on the physical console of a Swarm node provides options to administer and troubleshoot a node and hardware.



Actions from the System Menu:

- Review system information
- Diagnose startup errors
- Format hard disks
- Perform diagnostics
- Shut down and restart the cluster nodes



Caution

Use the option to reformat all devices after verification. With the System Menu, *all* discovered devices can be reformatted, *including* the Swarm USB flash drive. Verify it is *removed* before choosing this option.

Verify the node and/or cluster configuration file is configured before beginning.



Disabling for Security

Lock down access to the System Menu on the console if this node is located where it is not physically secure and limited to Swarm administrators. (v10.2.1)

Enable this option in the node.cfg file for this node:

```
security.securePhysicalConsole = true
```

Menu Options

The node has been secured against unauthorized tampering if these menu options are not accessible on the console. This option is controlled by the [Node-level configuration setting](#), `security.securePhysicalConsole`.

Menu	Usage
------	-------

<p>System Information</p>	<p>1. Configuration - Displays the current node configuration. Press : q to close.</p> <p>2. Startup Errors - Displays the boot errors. This option displays only if errors appeared during boot-up.</p>
<p>Disk Volumes</p>	<p>Displays a list of all detected volumes of at least 8 GB in size.</p> <p>Select a volume and enter one of the following:</p> <ul style="list-style-type: none"> • F to format the volume • I to identify the volume by flashing its LED • ALL to perform F and I on all disks <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p> Caution</p> <p>All discovered devices are reformatted, including the Swarm USB flash drive if selecting ALL and press F. Remove the USB flash drive before choosing this option.</p> </div>
<p>Diagnostics</p>	<p>Assists with gathering network and log information and provides utilities such as <code>ping</code> and <code>traceroute</code>.</p> <p>Select an option from the menu and follow the prompts on the screen to complete the tasks.</p> <ol style="list-style-type: none"> 1. Kernel log (dmesg) 2. Systemd Unit Status 3. Systemd journal 4. Time synchronization (NTP) 5. Top processes list 6. Network Interfaces and Routes 7. Ping a host 8. Traceroute to a host 9. Socket connections 10. Network send/receive buffers 11. ARP cache
<p>System Control</p>	<p>Enables node management:</p> <ol style="list-style-type: none"> 1. Reboot System - Restart the node. 2. Shutdown System - Shut down the node (stop all processes and power it off). 3. Stop Storage Processes - Stops the running Swarm processes so volumes can be reformatted if needed.

Formatting a Disk

The disk automatically format on bootup (or insertion, in the case of hot plug) if a disk has not been recognized by Swarm and it boots into a Swarm chassis. Use the System Menu on the storage node if a disk has been retired from a cluster and needs to be reformatted.

 **Note**

This process requires the chassis to be taken offline.

1. Using a remote terminal or a keyboard and monitor, access the **System Menu** on the Swarm storage node.
2. Take the chassis offline.
 - a. Select **4. System Control**.
 - b. Select **3. Stop Storage Processes**.
 - c. Verify by choosing **Yes**. At this point, **Storage Processes: STOPPED** displays.
3. Mark the disk for reformatting on next reboot.
 - a. Select **2. Disk Volumes**
 - b. Arrow down to the disk in question and press "E"
 - c. When prompted, type `FORMAT` to continue. (This only queues the formatting, so do not expect a delay.)
4. Restart the chassis.
 - a. Select **4. System Control**.
 - b. Select **1. Reboot System**.

Reformatting all Disks in a Cluster



Note

Reformat one node at a time to reformat an entire cluster. The remaining nodes in the cluster begin replicating and moving objects to the reformatted node if this is not performed.

To reformat all disks in a cluster:

1. Shut down all nodes in the cluster.
2. Remove the USB flash drive from any node in the cluster.
3. (*versions prior to 6.0*) Edit the configuration file and remove the `cluster.settingsUuid` value.
4. Insert the USB flash drive in the node and reboot it.
5. Select **Disk Volumes** in the System Information menu.
6. Select a formatting option in the Disk Volumes menu.
7. Wait for the selected disks to be formatted.
8. Shut down the node.
9. Repeat step 2 through step 8 on the remaining cluster nodes, one node at a time.
10. Start the nodes in any order after all nodes are reformatted.

Rebooting the Storage Cluster

Verify the updated USB flash drives, configuration server, or CSN is prepared and begin the reboot process after the cluster is shut down.

This is the recommended power-on sequence:

1. Start a single node.
2. Verify the node boots properly. This process verifies that the software loads correctly and the node can communicate with the network.
3. Repeat on each remaining node in the cluster.

When initially booting a node, the `hpStartDelay` parameter provides a 15-minute delay window by default for all nodes to boot up and join the cluster before Swarm begins to check for missing cluster nodes. As long as all nodes in the cluster are running within this window, failed volume recovery (FVR) and erasure-coded recovery (ECR) operations can be avoided.

This window exists immediately following a node reboot. After 15 minutes, the recovery operations begin immediately after Swarm detects a missing node.

Restoring a previous version

The software can usually be reverted to a previous version using the same method performed during an upgrade if a legacy software version on any node needs to be installed.

Limitations

- Reverting to a previous version is not supported when upgrading to version 6.0 or later from a version prior to 6.x due to internal Swarm changes.
- The 6.5.1 release can be downgraded to any 6.x release *except* 6.5.0.

Initializing a Storage Cluster

This section describes how to initialize a Swarm node.

- [Editing the Configuration File](#)
- [Configuring the Storage Volumes](#)
- [Configuring the DHCP or Static Network](#)
- [Setting the Admin Passwords](#)
- [Booting Swarm Nodes](#)

i **Platform Server**

Skip this section if using [Platform Server](#): these tasks are already performed.

Follow the readme file in the software distribution that describes how to set up the drive and what to copy over from the software distribution to set up a new USB flash drive with Swarm. Those steps must be completed before beginning edits of the **node.cfg** on the USB drive.

i **Warning**

Verify no one (including IT personnel and any on-site contractors) reboots a non-Swarm system while a Swarm-configured USB is mounted. The drives can be reformatted, with permanent data loss if a computer accidentally boots from a Swarm USB.

Editing the Configuration File

This section describes how to configure a storage cluster node by manually editing the node and/or cluster configuration file.

Note

A fail-safe timer mechanism is included in the Swarm startup process that restarts the boot process if the boot error screen displays for more than 15 minutes. This reboot is intended to compensate for temporary network loss and is canceled if a keyboard is used on the console.

Manually edit the node.cfg file:

1. Open the .cfg file on the USB flash drive or PXE configuration server in a text editor.
2. Set the setting **disk.volumes = all**. This setting allows Swarm use of all available volumes on the node.
3. Set the **cluster.name** configuration setting to the name of the cluster. Use an [IANA-compatible domain name](#), such as `cluster.example.com`.
Important – Configure all nodes in the cluster with the *same cluster name*.
4. Modify the network configuration to assign node IP addresses if DHCP is *not* being used.
5. Set other [configuration settings](#) as desired.
6. Save and close the .cfg file.
7. Verify the USB flash drive is safely unmounted or stopped if a node boots from a USB flash drive; otherwise, changes are not saved to the .cfg file.
8. Verify a valid **license.txt** file is located in the **caringo** directory.
9. Set up and configure the NTP server. (See [Configuring an External Time Server](#).)
10. Set up and configure the syslog server. (See [Configuring an Rsyslog Server](#).)
11. Boot the Swarm node.

Important

Multiple reboots in a row indicate there may be a problem with the system configuration. Check the syslog server for errors if this occurs.

Configuring the Storage Volumes

Swarm reads the `disk.volumes` setting from the node and/or cluster configuration files to determine which disks can be used for content storage. Swarm is pre-configured with a dummy value that prevents the disks on the cluster nodes from being reformatted during the installation procedure. An administrator must edit the `disk.volumes` setting value with the keyword **all** or with a list of disks Swarm can use for storage after Swarm is installed.

The easiest way to use all disks in a node for content storage is to use the keyword **all**:

```
disk.volumes = all
```

Swarm automatically excludes the Swarm USB flash drive from being used for storage when using the `all` keyword.

Important

[Hot plugging](#) is only supported for `disk.volumes = all`.

Set `disk.volumes` to a space-separated list of drive identifiers to set the disk device names.

Swarm uses standard Linux volume identifiers such as `/dev/sda` and `/dev/sdb`. Use the [Swarm System Menu](#) or access the Swarm node using another Linux system if volume identifiers cannot be identified.

Using the preceding example, set `disk.volumes` to:

```
disk.volumes = /dev/sda /dev/sdb
```

Warning

Verify no one (including IT personnel and any on-site contractors) reboots a non-Swarm system while a Swarm-configured USB is mounted. If a computer accidentally boots from a Swarm USB, the drives can be reformatted, with permanent data loss.

Configuring the DHCP or Static Network

The easiest way to set up the cluster is to configure DHCP to automatically assign an IP address to each cluster node. DHCP is used by default in a CSN configuration. A DHCP server is needed in the network connected to the cluster nodes. Swarm can be booted from the USB flash drive or network environment when completed.

Edit the **network.ipAddress**, **network.netmask**, and **network.gateway** settings in the node and/or centralized configuration files to assign static IP addresses to the nodes. Set *all three settings* when using the static IP address configuration option. In a centralized configuration environment, these settings must be set in a custom configuration file on each cluster node because this information cannot be shared between the nodes.

Edit the IP address, network mask, and default gateway settings on the nodes:

Network settings

```
network.ipAddress = 10.20.30.101
network.netmask = 255.255.255.0
network.gateway = 10.20.30.1
```

1. Open the centralized configuration file or open the configuration file on the first node.
2. Edit the network settings in the file.
3. Repeat for each node if editing the node configuration file. No additional changes are required if editing the centralized configuration file.

Setting the Admin Passwords

- [Granting Swarm Access](#)
 - [Disabling SNMP](#)
 - [Caution](#)
- [Encrypting Passwords](#)
- [Updating Passwords](#)
 - [Caution](#)
 - [Swarm has never booted](#)
 - [Updating SNMP passwords](#)
 - [Updating Swarm admin password](#)
 - [Changing admin password](#)

Granting Swarm Access

Swarm uses two pairs of security lists to grant access to storage cluster management and viewing:

- **Administrators** can access the Swarm UI and change the cluster configuration. SNMP *read/write* access is handled separately.
- **Operators** can view the Swarm UI. SNMP *read-only* access is handled separately.



Disabling SNMP

Disable the Swarm Storage setting `snmp.enabled` if SNMP needs to be disabled cluster-wide, such as for a security need or using Swarm in containers. (v12.0)

Each user list is specified by a [configuration parameter](#) with name/value pairs in the Swarm Storage configuration file (`cluster.cfg` (CSN) or else `node.cfg`). Those passwords needed for SNMP access are handled in separate settings (v10.0):

```
security.administrators = {'admin':'adminpassword','admin2':'adminpassword2'}
security.operators = {'operator':'operatorpassword','operator2':'operatorpassword2'}
snmp.roCommunity = public
snmp.rwCommunity = ourpwdofchoicehere
```

or section notation:

```
[security]
administrators = {'admin':'adminpassword','admin2':'adminpassword2'}
operators = {'operator':'operatorpassword','operator2':'operatorpassword2'}
```

```
[snmp]
roCommunity = public
rwCommunity = ourpwdofchoicehere
```

Setting name	Default	Notes
--------------	---------	-------

security.administrators	<pre>{ 'admin': 'ourpwdofchoicehere' }</pre>	<p>One or more username:password pairs. Sets credentials for who can administer the cluster via the Swarm UI.</p> <p>Upgrading from 9.x – Remove the <code>snmp</code> username from here and update <code>snmp.rwCommunity</code> with the password if the value includes the <code>snmp</code> username.</p> <ul style="list-style-type: none"> • Example: <code>{ 'admin': 'adminpassword', 'admin2': 'adminpassword2' }</code>
security.operators	<pre>{ }</pre>	<p>One or more username:password pairs. Sets credentials for who can view the Swarm UI.</p> <p>Upgrading from 9.x – It is ignored if the value includes an <code>snmp</code> username; remove it from here and update <code>snmp.roCommunity</code> with the password.</p> <ul style="list-style-type: none"> • Example: <code>{ 'operator': 'operatorpassword', 'operator2': 'operatorpassword2' }</code>
snmp.rwCommunity	<pre>ourpwdofchoicehere</pre>	<p>String. Password for the SNMP read-write community.</p> <p>Required – The SNMP read-write password must be known to dynamically change the Swarm 'admin' password via SNMP. The config file <i>must</i> be edited to change the SNMP read-write password. The SNMP password is the sole option if the admin-level credentials are lost.</p>
snmp.roCommunity	<pre>public</pre>	<p>String. Password for the SNMP read-only community.</p>

Caution

- The name `admin` is reserved, so do not delete it, which can cause errors and affect performance. Define a complex password for protection if deciding not to use `admin`.
- Swarm prevents cluster booting if the SNMP security administrator (read/write user) is not set properly in the configuration file.
- All administrative users and passwords must agree on *all nodes* or certain cluster actions fail.
- Password updates are not complete until they are persisted in the cluster settings file across all nodes, and rapid, successive updates cannot be accepted on a given node until the first update completes processing.
- Change passwords from the **defaults** *before* putting the cluster in to production, and improve security by encrypting the Swarm passwords. *See next.*

Encrypting Passwords

Represent the password as a hexadecimal-encoded [MD5](#) hash of the following string instead of a clear text password:

```
username:user-list-name:password
```

Where username and password consist of ASCII characters and `user-list-name` can be either "CASstor administrator" or "CASstor operator".

To create the MD5 hash, use a programming language or a utility such as [md5sum](#) or Apache [htdigest](#). To update a node or cluster configuration file with a password hash created using `htdigest`:

1. Create a file containing a hash of the user name, password, and user list name:

```
htdigest -c password-file.txt "CAStor administrator" Jo.Jones
```

2. Enter and verify the user's password when prompted by **htdigest**.
3. Open the new file (`password-file.txt`) in a text editor. The hash is the *last* entry in the string:

```
Jo.Jones:CAStor administrator:08b0468c1d957b7bac24463dd2191a2d
```

Updating Passwords

The list of Administrators and passwords may be modified without rebooting by using several read-write SNMP OIDs. New administrative users can be added and existing users modified with the `addModifyAdministrator` SNMP OID. These are the essential commands:

- **Add admin users** – Include the new user name and password separated by a colon:
`addModifyAdministrator = "Jo.Jones:password1"`
- **Update password** for an existing user – Include the existing user name and new password separated by a colon:
`addModifyAdministrator = "Jo.Jones:password2"`
- **Delete admin users** (except the default admin and snmp users) – Send the name of an admin user:
`removeAdministrator = "Jo.Jones"`



Caution

- All administrative users and passwords must agree across *all nodes* or certain cluster actions fail.
- Any changes made via SNMP against a running cluster must be made in the node/cluster configuration file so any nodes offline when the change is made or new nodes added to the cluster after the fact can correctly authenticate cluster-wide actions.
- It can take several minutes for these SNMP changes to propagate in the cluster. During this update window, old passwords and deleted users continue to work for up to 10 minutes.

Important: How passwords are updated depends on which ones need updating and whether Swarm has ever been started.

Process	Examples and notes
---------	--------------------

<p>Swarm has never booted</p> <ol style="list-style-type: none"> 1. Create and hash an admin password. 2. Update passwords in the config file. 3. Important: Unmount/stop the USB drive or else the changes are not saved if booting from a USB flash drive. 4. Boot the Swarm cluster. 5. the password can be removed from the config file after the cluster is running. 	<p>Hash of password</p> <pre>\$ echo -n 'admin:CAStor administrator:NEWPASSWORD' md5sum cut -d ' ' -f1 7fe563b8532b3a460def0895895eebf5</pre> <p>The first time the cluster is booted the Swarm admin password <i>must</i> be in the config file:</p> <pre>[security] administrators = {'admin':'7fe563b8532b3a460def0895895eebf5'}</pre> <p>When the cluster is running, Swarm stores the admin password in the persisted Settings object, at which point it is safe to remove the password from the configuration file for security purposes:</p> <pre>[security] administrators = {}</pre>
<p>Updating SNMP passwords</p> <ol style="list-style-type: none"> 1. Update passwords in the config file. 2. Reboot the Swarm cluster. 	<p>Important – The SNMP read-write password must be known to dynamically change the Swarm 'admin' password. The config file <i>must</i> be edited if the SNMP read-write password needs to be changed..</p> <p>Proceed to change the Swarm 'admin' password after rebooting with the new SNMP password in the file</p>

<p>Updating Swarm admin password</p> <ol style="list-style-type: none"> 1. Create and hash an admin password. 2. Update password via SNMP, which Swarm saves in the persisted Settings object. 	<p>Changing admin password</p> <pre>snmpset -v2c -c SNMP- password -m +CARINGO-CASTOR-MIB SWARM-NODE-IP addModifyAdministrator s "admin:new- password"</pre> <pre>snmpset -v2c -c ourpwdofchoicewhere -m +CARINGO-CASTOR-MIB 172.20.3.85 addModifyAdministrator s "admin:7fe563b8532b3a460def0895895eebf5"</pre>
---	---

Frequently asked questions:

- *How do I change the active SNMP read-write password?* The SNMP passwords cannot be changed dynamically. Changing one or both requires a config file update and a cluster reboot.
- *What is the SNMP read-only password?* The read-only password 'public', which is the 'community string'
- *Is the read-only SNMP password in the persisted Settings object?* No
- *Can my SNMP read-write passwords in the persisted Settings object and cluster.cfg be different?* Yes, but the config file SNMP read-write password is used.
- *How do I change my admin password?* Update the password using SNMP and then update it in the config file unless it is removed from there.
- *How do I change my SNMP read-only password to the cluster?* Change the `snmp.roCommunity` setting in the config file and reboot the cluster.

Booting Swarm Nodes

Verify the basic configuration requirements are completed and the USB flash drive is inserted into the USB port on the cluster node if booting nodes from a USB flash drive. The Swarm operating system loads into system RAM and the `caringo\node.cfg` file is read from the USB flash drive after the hardware self-test is completed.

Verify the network boot environment is configured and online if booting the nodes from a network. Boot the nodes when completed.

A monitor may be attached to the node, but is not required.

Swarm Passwords

- [Granting Swarm Access](#)
 - [Disabling SNMP](#)
 - [Caution](#)
- [Encrypting Passwords](#)
- [Updating Passwords](#)
 - [Caution](#)
 - [Swarm has never booted](#)
 - [Updating SNMP passwords](#)
 - [Updating Swarm admin password](#)
 - [Changing admin password](#)

Granting Swarm Access

Swarm uses two pairs of security lists to grant access to storage cluster management and viewing:

- **Administrators** can access the Swarm UI and change the cluster configuration. SNMP *read/write* access is handled separately.
- **Operators** can view the Swarm UI. SNMP *read-only* access is handled separately.



Disabling SNMP

Disable the Swarm Storage setting `snmp.enabled` if SNMP needs to be disabled cluster-wide, such as for a security need or using Swarm in containers. (v12.0)

Each user list is specified by a [configuration parameter](#) with name/value pairs in the Swarm Storage configuration file (`cluster.cfg` (CSN) or else `node.cfg`). Those passwords needed for SNMP access are handled in separate settings (v10.0):

```
security.administrators = {'admin':'adminpassword', 'admin2':'adminpassword2'}
security.operators = {'operator':'operatorpassword', 'operator2':'operatorpassword2'}
snmp.roCommunity = public
snmp.rwCommunity = ourpwdofchoicehere
```

or section notation:

```
[security]
administrators = {'admin':'adminpassword', 'admin2':'adminpassword2'}
operators = {'operator':'operatorpassword', 'operator2':'operatorpassword2'}

[snmp]
roCommunity = public
rwCommunity = ourpwdofchoicehere
```

Setting name	Default	Notes
--------------	---------	-------

security.administrators	<pre>{ 'admin': 'ourpwdofchoicehere' }</pre>	<p>One or more username:password pairs. Sets credentials for who can administer the cluster via the Swarm UI.</p> <p>Upgrading from 9.x – Remove the <code>snmp</code> username from here and update <code>snmp.rwCommunity</code> with the password if the value includes the <code>snmp</code> username.</p> <ul style="list-style-type: none"> • Example: <code>{ 'admin': 'adminpassword', 'admin2': 'adminpassword2' }</code>
security.operators	<pre>{ }</pre>	<p>One or more username:password pairs. Sets credentials for who can view the Swarm UI.</p> <p>Upgrading from 9.x – It is ignored if the value includes an <code>snmp</code> username; remove it from here and update <code>snmp.roCommunity</code> with the password.</p> <ul style="list-style-type: none"> • Example: <code>{ 'operator': 'operatorpassword', 'operator2': 'operatorpassword2' }</code>
snmp.rwCommunity	<pre>ourpwdofchoicehere</pre>	<p>String. Password for the SNMP read-write community.</p> <p>Required – The SNMP read-write password must be known to dynamically change the Swarm 'admin' password via SNMP. The config file <i>must</i> be edited to change the SNMP read-write password. The SNMP password is the sole option if the admin-level credentials are lost.</p>
snmp.roCommunity	<pre>public</pre>	<p>String. Password for the SNMP read-only community.</p>

Caution

- The name `admin` is reserved, so do not delete it, which can cause errors and affect performance. Define a complex password for protection if deciding not to use `admin`.
- Swarm prevents cluster booting if the SNMP security administrator (read/write user) is not set properly in the configuration file.
- All administrative users and passwords must agree on *all nodes* or certain cluster actions fail.
- Password updates are not complete until they are persisted in the cluster settings file across all nodes, and rapid, successive updates cannot be accepted on a given node until the first update completes processing.
- Change passwords from the **defaults** *before* putting the cluster in to production, and improve security by encrypting the Swarm passwords. *See next.*

Encrypting Passwords

Represent the password as a hexadecimal-encoded [MD5](#) hash of the following string instead of a clear text password:

```
username:user-list-name:password
```

Where username and password consist of ASCII characters and `user-list-name` can be either "CASstor administrator" or "CASstor operator".

To create the MD5 hash, use a programming language or a utility such as [md5sum](#) or Apache [htdigest](#). To update a node or cluster configuration file with a password hash created using `htdigest`:

1. Create a file containing a hash of the user name, password, and user list name:

```
htdigest -c password-file.txt "CAStor administrator" Jo.Jones
```

2. Enter and verify the user's password when prompted by **htdigest**.
3. Open the new file (`password-file.txt`) in a text editor. The hash is the *last* entry in the string:

```
Jo.Jones:CAStor administrator:08b0468c1d957b7bac24463dd2191a2d
```

Updating Passwords

The list of Administrators and passwords may be modified without rebooting by using several read-write SNMP OIDs. New administrative users can be added and existing users modified with the `addModifyAdministrator` SNMP OID. These are the essential commands:

- **Add admin users** – Include the new user name and password separated by a colon:
`addModifyAdministrator = "Jo.Jones:password1"`
- **Update password** for an existing user – Include the existing user name and new password separated by a colon:
`addModifyAdministrator = "Jo.Jones:password2"`
- **Delete admin users** (except the default admin and snmp users) – Send the name of an admin user:
`removeAdministrator = "Jo.Jones"`



Caution

- All administrative users and passwords must agree across *all nodes* or certain cluster actions fail.
- Any changes made via SNMP against a running cluster must be made in the node/cluster configuration file so any nodes offline when the change is made or new nodes added to the cluster after the fact can correctly authenticate cluster-wide actions.
- It can take several minutes for these SNMP changes to propagate in the cluster. During this update window, old passwords and deleted users continue to work for up to 10 minutes.

Important: How passwords are updated depends on which ones need updating and whether Swarm has ever been started.

Process	Examples and notes
---------	--------------------

<p>Swarm has never booted</p> <ol style="list-style-type: none"> 1. Create and hash an admin password. 2. Update passwords in the config file. 3. Important: Unmount/stop the USB drive or else the changes are not saved if booting from a USB flash drive. 4. Boot the Swarm cluster. 5. the password can be removed from the config file after the cluster is running. 	<p>Hash of password</p> <pre>\$ echo -n 'admin:CAStor administrator:NEWPASSWORD' md5sum cut -d ' ' -f1 7fe563b8532b3a460def0895895eebf5</pre> <p>The first time the cluster is booted the Swarm admin password <i>must</i> be in the config file:</p> <pre>[security] administrators = {'admin':'7fe563b8532b3a460def0895895eebf5'}</pre> <p>When the cluster is running, Swarm stores the admin password in the persisted Settings object, at which point it is safe to remove the password from the configuration file for security purposes:</p> <pre>[security] administrators = {}</pre>
<p>Updating SNMP passwords</p> <ol style="list-style-type: none"> 1. Update passwords in the config file. 2. Reboot the Swarm cluster. 	<p>Important – The SNMP read-write password must be known to dynamically change the Swarm 'admin' password. The config file <i>must</i> be edited if the SNMP read-write password needs to be changed..</p> <p>Proceed to change the Swarm 'admin' password after rebooting with the new SNMP password in the file</p>

<p>Updating Swarm admin password</p> <ol style="list-style-type: none"> 1. Create and hash an admin password. 2. Update password via SNMP, which Swarm saves in the persisted Settings object. 	<p>Changing admin password</p> <pre>snmpset -v2c -c SNMP- password -m +CARINGO-CASTOR-MIB SWARM-NODE-IP addModifyAdministrator s "admin:new- password"</pre> <pre>snmpset -v2c -c ourpwdofchoicewhere -m +CARINGO-CASTOR-MIB 172.20.3.85 addModifyAdministrator s "admin:7fe563b8532b3a460def0895895eebf5"</pre>
---	---

Frequently asked questions:

- *How do I change the active SNMP read-write password?* The SNMP passwords cannot be changed dynamically. Changing one or both requires a config file update and a cluster reboot.
- *What is the SNMP read-only password?* The read-only password 'public', which is the 'community string'
- *Is the read-only SNMP password in the persisted Settings object?* No
- *Can my SNMP read-write passwords in the persisted Settings object and cluster.cfg be different?* Yes, but the config file SNMP read-write password is used.
- *How do I change my admin password?* Update the password using SNMP and then update it in the config file unless it is removed from there.
- *How do I change my SNMP read-only password to the cluster?* Change the `snmp.roCommunity` setting in the config file and reboot the cluster.

Licensing Swarm

A license based on the contractually agreed amount of storage space purchased for the cluster is received when Swarm is purchased. The `license.url` parameter in the node or cluster configuration file indicates the location of the license file.

Important

Swarm licenses are based on raw usage, the total footprint on disk. Increasing the number of replicas or EC parity segments increases the footprint, which has the effect of decreasing the net space the license supports.

Swarm performs the following to evaluate the licensing:

1. Swarm starts with the default trial license, which is 2 TB per cluster for non-CSN clusters.
2. Swarm attempts to read the license file from the location specified by `license.url`.
The license file is `/caringo/license.txt` on the node's USB flash drive or in the configuration file on the web or FTP server by default.
3. Swarm uses the file specified by `license.url` if valid. The storage capacity is set to 0 if invalid.

Note

Swarm uses the last valid license if Swarm fails to read the license specified by `license.url` (the file is on a web server that is temporarily unavailable).

- [Sample License](#)
- [Changing the license.url setting](#)
- [Cluster Capacity Monitoring](#)
- [Troubleshooting licenses](#)
- [Upgrading a License](#)

Sample License

Below is a sample of a Swarm license.

```

----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1
# #####
# Swarm License File
# License S/N: 200804261512-8402
# Generated By: Eric Smith
# Comments:
# #####
licenseFormat = 1.1
cn = ACME Widgets, Inc.
street = 123 Street A, Building #23
cl = Austin
st = Texas
postalCode = 78746
co = USA
clusterDescription = Corporate Office
# License Components
expirationDate = none
featureClusterMaxTB = 2
featureContentIndexing = no
featureErasureCoding = yes
featureMinimumMinReps = 1
----BEGIN PGP SIGNATURE---
Version: GnuPG v1.4.6 (GNU/Linux) iD8DBQFIHztDRYikRJU1RfMRAusHAKCX9ABhEBgQz/TyTy+gT5gXf7hNmQCeKxL
----END PGP SIGNATURE-----

```

Changing the `license.url` setting

You can specify alternate file names and locations using the `license.url` option in the node or cluster (Platform Server) configuration file. DNS names for FTP and HTTP hosts are supported as long as the DNS server and domain information is set by DHCP or is in the node and/or cluster configuration files. Errors processing the `license.url` setting are visible on the Linux system console during boot-up, but they do not prevent a successful boot.

Example `license.url` configurations:

```
license.url=http://192.168.0.103/license.txt
license.url=ftp://myftpserver/storagecluster_license.txt
license.url=file:///caringo/customerlicense.txt
```



Important

If you change the name of the license file for the local USB drive, the file must still remain in the `caringo` directory or one of its subdirectories.

Cluster Capacity Monitoring

Important

The maximum storage space in a cluster is the *lesser of the two*: total physical space and licensed capacity.

Note

The `featureMinimumMinReps` item in the license file is unrelated to the number of replicas required to be kept in the cluster (specified in the configuration), so it does not affect licensed capacity.

Running low on space – Two settings help stay ahead of running out of cluster capacity:

- `console.spaceWarnLevel` specifies what percentage of space remaining on the node triggers a warning. Set to 25% of available space by default.
- `console.spaceErrorLevel` specifies what percentage of space remaining on the node triggers an alert. Set at 10% of available space by default.

Within an hour of reaching either of these thresholds, an announcement appears both in the logs and the UI.

Out of space – All inbound write requests are refused when available space in the cluster reaches zero. Internal replication and relocation continue and all previously stored data is still readable. No data is lost or corrupted.

Contact a DataCore Sales or Support representative to increase licensed capacity.

Troubleshooting licenses

Attempt the following if you experience issues with your license:

- Verify the license specified by `license.url` is a valid Swarm license if the capacity of a node in your cluster is 0. Verify the license has not been manually edited as this invalidates the license key.

Note

Swarm checks every 15 minutes for license updates. The electronic signature in the license becomes invalid modified, causing Swarm to revert to the default license.

- Verify the following if the capacity of a node is set to the default of 2 TB for an extended period of time:
 - Using a valid Swarm license.
 - `license.url` is set the to the location of the license.
 - The location specified is available to the cluster. Swarm uses the default 2 TB license because it was the last known valid license if `license.url` is set to a location unavailable to the cluster soon after it was booted.

Upgrading a License

An updated Swarm license file may be required when adding additional storage capacity, a new feature, or updated client information.

All new license files must be issued by a Sales representative to verify they have an electronic signature recognized and approved by Swarm.



Important

Match the path and filename defined in the [license.url setting](#) in the configuration when updating the license file on a running cluster.

How to update the license

The default location is the **caringo** directory of the local node's USB drive or centralized configuration server.

- Upload the license using the [CSN Console](#) if running CSN 8.3.
- Upload the license using either the Swarm UI ([Settings > License](#)) or the CLI [platform upload license](#) command if running Platform Server.

How Swarm validates the license

Swarm checks the license file every 15 minutes.

- Swarm validates the license and updates the client information and/or licensed capacity as necessary if the file is updated. An announcement appears in the syslog and the UI on success.
- The cluster nodes report a critical error in the syslog and the UI if a license update fails to validate a new license file.
- Swarm continues to use the previous license file until the validation error is corrected if an update fails with a validation error.
- Swarm analyzes the file for updates when it becomes available if the license file is located on an HTTP server not available when Swarm starts. Swarm may publish a corresponding update announcement in the syslog and the UI, even if the file itself has not changed.

Implementation Checklist

Here is a top-level view of typical issues to resolve before launching a production implementation of a Swarm cluster. These topics are covered in the sections that follow.

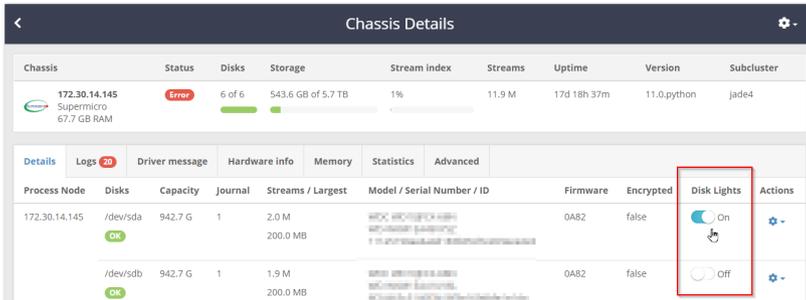
Hardware	<ul style="list-style-type: none"> Swarm nodes meet minimum requirements: <ul style="list-style-type: none"> CPU: Intel or AMD, 64-bit, 2+ cores RAM: Sufficient memory available for anticipated object count Drives: SATA, SAS, or SCSI – enterprise-class with 24x7 duty cycle Network: 1000 Mbps or faster (100 Mbps minimum) – multiple interfaces if redundancy required Number of nodes is greater than the number of configured replicas and EC $k+p$ Sufficient free space in the cluster to recover content if a node fails?
Booting Type	<ul style="list-style-type: none"> Platform Server, or Self-boot options: <ul style="list-style-type: none"> USB booting? Verify configuration files are the same. PXE/network booting? Centralized configuration files?
Network Management	<ul style="list-style-type: none"> DHCP or static IP addressing? NTP server is configured and available. Syslog server is configured and available; configured local6* and log rotation policy. Network filters are configured to verify authorized users can reach the cluster. DNS names for nodes are set up. Network switch redundancy is sufficient for your deployment. Monitoring procedures are in place: SNMP or assigned administrator watching console. Verified bandwidth to cluster nodes. Defined Swarm bonding mode (active-backup, balance-alb, or 802.3ad).
Storage Config	<ul style="list-style-type: none"> The <code>administrator</code> and <code>operator</code> default passwords are updated to a new, secure value. The node and/or cluster configuration file is updated and validated.
Data Protection	<ul style="list-style-type: none"> Does the default number of replicas (<code>policy.replicas default</code>) meet your protection requirements? Does the default EC ratio (<code>policy.ecEncoding</code>) meet your protection requirements? Set lifepoints 'reps' for necessary file types? File retention period and deleteability (yes or no) defined in lifepoints? Any non-deletable files or automatic delete lifepoints? UUID management mechanism protected (database or other means) if not using metadata search?
Default Domain	<ul style="list-style-type: none"> Set <code>cluster.name</code> to your desired default domain name, a valid IANA FQDN host name (<code>cluster1.cloud.example.com</code>) Create a domain that matches the value of <code>cluster.name</code>. This is the default domain for this cluster. Set <code>cluster.enforceTenancy = True</code> to verify unnamed content is homed in a default domain if no domain is specified

<p>Application Setup</p>	<ul style="list-style-type: none"> • What storage protocols do your client applications use? • If using HTTPS (SSL/TLS), create appropriate X.509 certificates and install on encryption off-load hardware/software. • SCSP clients: <ul style="list-style-type: none"> • Verify relevant custom metadata is defined in natively integrated client. • How does your application access the primary access node (PAN)? • Create user accounts and, optionally, authentication tokens. • S3 clients: <ul style="list-style-type: none"> • Create S3 authentication tokens for applications • Method for directing SCSP and S3 requests to appropriate Gateway port?
<p>Testing</p>	<ul style="list-style-type: none"> • Conduct end-to-end testing of the exact hardware, network setup, and software version to be used in production to verify there are no compatibility issues. • Test functionality using all client protocols.
<p>Maintenance</p>	<ul style="list-style-type: none"> • What is your disk drive replacement strategy? • What are your maintenance windows for software updates? • Do you use staging or testing environments to pre-qualify updates?

Drive Identification Plugin

- [Best practice](#)
- [Disk Light Customization](#)
- [Disk Light API](#)

Swarm supports a disk identification function that flashes an LED light next to a volume you select in the Swarm UI (or legacy Admin Console). The plugin turns on the disk light when a disk fails to mount, whether at reboot or hot-plug event.



The **Identify** function is implemented as a Linux shell script daemon. When the **Identify** function is enabled, Swarm attempts one of several tools (sg_ses, MegaCli64, sas2ircu, sas3ircu) to turn on the identification light. If none of those tools succeed, Swarm continuously reads blocks from the selected disk to force the I/O activity disk light to flash.

- The script is provided in the Swarm distribution: `samples/genericDrivePlugin.sh`

- In the Swarm bundle you downloaded (Swarm-VERSION-DATE.zip), expand the Storage files (Storage/Storage-VERSION-ARCH.zip).
- The script is overridden if there is a custom value in the [disk.volumelIdentifyFiles](#) setting.

i Best practice

Use the generic plugin and remove any [disk.volumelIdentifyFiles](#) value from your configuration files.

Disk Light Customization

Integrators or hardware vendors with knowledge of manufacturer-specific methods for flashing disk light may substitute an alternative method using the plugin disk identification API.

To customize disk light script:

1. Open the Swarm software distribution and navigate to the `samples` directory.
2. In the directory, copy the `genericDrivePlugin.sh` script to a new directory.
3. Rename the script with a unique name that does not contain any periods or special characters.
4. Open the script and modify the `identifyOn()` and `identifyOff()` functions as required to implement an alternate disk light identification mechanism.
5. Archive (**tar**) and compress (**gzip**) the script and any other required custom files.
6. Deploy the script.
7. Place the tar file on an HTTP server or Swarm USB drive.
8. Update the [disk.volumelIdentifyFiles](#) setting to point to the location of your tar file.

Disk Light API

These are the key functions in the generic script:

identifyOn()	Turns on the disk identification light.
identifyOff()	Turns off the disk identification light.
_monitorAndInitiate()	Polls for a disk status file and initiates identification by spawning <code>_identifyOn()</code> .
_identifyOn()	Spawned as a process when identification is turned on by Swarm for a volume (for example, one process per volume). It calls the <code>_identifyOn()</code> function and then waits for identify to get turned back off by Swarm so it can call the <code>_identifyOff()</code> function.

Configuring Swarm for Gateway

This section provides information specific to running Swarm Storage with Gateway. Install and configure Swarm, the storage cluster (storage nodes running on dedicated hardware) before proceeding.

- [Network Placement](#)
 - [Caution](#)
- [Domain Management](#)
- [Elasticsearch Servers](#)
 - [Listing consistency](#)
- [Configuration Requirements](#)
 - [Caution](#)
 - [Note](#)

Network Placement

When deployed with Gateway, the storage nodes should be placed on a network subnet not directly accessible to client applications. All user communications with the storage cluster must go through the Gateway.



Caution

If users are allowed to communicate directly with the storage cluster nodes, they may bypass access security, the business rules for content metadata, and audit logging performed by the Gateway and may render content in the cluster unusable to the Gateway. Only allow direct access to the storage cluster nodes under highly controlled circumstances, such as administrator-only operations or trusted applications.

Domain Management

The Swarm cluster provides for logical separation of content among multiple tenants through the use of storage domain names. Gateway has the following requirements beyond those for a baseline storage deployment and client usage.

- An administrative domain must be created in the storage cluster.
- Storage domains must adhere to IANA naming standards (valid DNS names).
- Client applications should specify a storage domain in every request (if not, the request goes to the default domain, with `enforceTenancy=True`).

The storage domain name for an operation is specified by the client application according to the following precedence from highest to lowest:

- SCSP `domain=X` query argument
- HTTP `X-Forwarded-Host` header
- HTTP `Host` header

Storage domains in Swarm must resolve to at least one IP address ("A" record) for client applications to make use of the Host header to identify the storage domain with most HTTP/1.1 libraries. The resolved IP address should be for a Gateway or some other front-end network appliance such as a load balancer if applicable. Using a DNS round-robin with IP addresses is a valid configuration to use if there are multiple Gateway servers.

This is an example of a *BIND 9* zone file implementing a wildcard of all storage domains within the `cloud.example.com` parent DNS domain and points them to the IP address `10.100.100.100`.

```
$TTL 600 @ IN SOA cloud.example.com. dnsadmin.example.com. (
  2016070201 ; Serial number
  4H      ; Refresh every 4 hours
  1H      ; Retry every hour
  2W      ; Expire after 2 weeks
  300 ) ; nxdomain negative cache time of 5 minutes
IN NS ns1.example.com.
* IN A 10.100.100.100
```

In the example zone file, `10.100.100.100` is the IP address used by client applications to communicate with the Gateway or a front-end load balancer. The names hydrogen2.cloud.example.com and oxygen.cloud.example.com both resolve to the same IP address.

Elasticsearch Servers

When using the S3 storage protocol, the metadata search service must be accessible to the Gateway servers.

When deployed with Gateway, like the storage nodes, the typical placement is on a network subnet not directly accessible to the client applications. There are no end-user supported API calls directly to the metadata search service.

i Listing consistency

Search feeds show eventual consistency as content changes, but enabling the [Gateway Configuration \[s3\]](#) option `enhancedListingConsistency` improves the search-after-create response to the client applications using the Gateway.

Configuration Requirements

Use these Swarm configuration settings and adhere to the following operational changes when using Swarm Storage with Gateway. These configuration changes refer to the configuration file(s) for Swarm.

- **CSN** – This is the cluster-wide file: `/var/opt/caringo/netboot/content/cluster.cfg`
- **Platform Server** – This is the cluster-wide file used to deploy, which is located by default here: `/etc/caringo/cluster.cfg`
- **No platform server** – This is the node-specific configuration file: `node.cfg`

i Caution

Failure to use these settings and operational changes can prevent Gateway from working properly with the storage cluster.

Requirement	Description
Optimize GETs	<p>With Swarm 12.0 and higher, a setting can be added to improve performance through Gateway. Enable <code>scsp.enableVolumeRedirects</code> to permit Gateway to redirect GET requests to volume processes. These redirects increase efficiency, especially with reading small objects.</p> <pre>scsp.enableVolumeRedirects = True</pre>

<p>Enable an EC encoding</p>	<p>S3 multipart (large file) writes fail if erasure coding is not configured; define an ecEncoding if using S3.</p> <pre>policy.ecEncoding = {k:p}</pre> <p>See Implementing EC Encoding Policy.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>the value put into the configuration file is used on first boot. <i>After</i> the cluster is running, the original values are persisted and must be updated dynamically (using UI or SNMP). See Persisted Settings (SNMP).</p> </div>
<p>Clear legacy settings</p>	<p><i>Unless needed for backwards compatibility</i> (because untenanted objects are used in the cluster and do not need S3), enable tenancy for unnamed objects, which verifies every object is written to a domain (see How enforceTenancy Works):</p> <pre>cluster.enforceTenancy = True</pre> <p>Set it to <code>True</code> and reboot the cluster if this was set to <code>False</code>.</p>
<p>Storage Domain Management</p>	<p>Only create and manage storage domains through the Content UI or programmatically through the Gateway's management API.</p> <p>The cluster configuration contains <code>security.noauth=False</code> if storage domain management displays in the legacy Admin Console (port 90), which is not supported by Content Gateway. Set it to <code>True</code> and reboot the cluster.</p> <p>Troubleshooting: If the Content UI reports "Page Not Found: The original bucket to which this collection refers cannot be found or has been replaced", it is likely the domain was created by the legacy Admin Console (port 90) and contains the legacy <code>Castor-Authorization</code> header. DataCore Support for help correcting the domain.</p>

SCS Implementation

- [Overview](#)
 - [Sample Environment](#)
- [Prerequisites](#)

Overview

Swarm Cluster Services (SCS) is a replacement for CSN but in a modernized form that is easy to install and maintain. It provides multiple services bundling them into a single package for booting and configuring Swarm storage.

Some of the major services that SCS provides are:

- PXE booting
- Configuration management
- Logging server
- DHCP
- NTP

Sample Environment

Below is an illustration of a sample deployment used as an example throughout the following pages.

Role	Application Network	Private Network	OS	vCPU	RAM (GB)	Disk Size (GB)	Remark
SCS	172.16.33.10	192.168.9.10	CentOS 7 / RHEL 7	4	8	100	Swarm Cluster Services
ES75-01	172.16.33.21	192.168.9.21	CentOS 7 / RHEL 7	4	8	100	Elasticsearch 7.5.2 Cluster (SSD)
ES75-02	172.16.33.22	192.168.9.22	CentOS 7 / RHEL 7	4	8	100	Elasticsearch 7.5.2 Cluster (SSD)
ES75-03	172.16.33.23	192.168.9.23	CentOS 7 / RHEL 7	4	8	100	Elasticsearch 7.5.2 Cluster (SSD)
GW77-01	172.16.33.16	192.168.9.16	CentOS 7 / RHEL 7	6	8	100	Cloud Gateway 7.7
GW77-02	172.16.33.17	192.168.9.17	CentOS 7 / RHEL 7	6	8	100	Cloud Gateway 7.7
Swarm Telemetry	172.16.33.15	192.168.9.15	CentOS 7 / RHEL 7	4	8	50	Swarm Telemetry – Gateway, Storage Nodes monitoring

Role	Virtual IP	Application Network	OS	vCPU	RAM (GB)	Disk Size (GB)	Remark
HAProxy01	172.16.33.90	172.16.33.91	CentOS 7	4	4	100	Reverse Proxy + Keepalived
HAProxy02		172.16.33.92	CentOS 7	4	4	100	Reverse Proxy + Keepalived

Prerequisites

- CentOS setup
- Swarm 14.0 bundle package
- Disable IGMP snooping on Private Network VLAN

Swarm Cluster Services (SCS) is distributed as an RPM. The installation steps for SCS are:

- [Setup RHEL/CentOS for SCS](#)
- [Swarm 14.0 Bundle Package](#)

- [SCS Installation](#)
- [Run the Initialization Wizard](#)
- [Add the Swarm Storage Component](#)
- [Finalize Swarm Configuration Settings](#)
- [Configure DHCP](#)
- [Install Swarm License](#)
- [Power on the Swarm Storage Nodes](#)
- [Setup Elasticsearch Cluster](#)
- [Swarm Gateway](#)
- [Domain-level Replication](#)
- [Swarm Telemetry](#)

Setup RHEL/CentOS for SCS

- [Required Version](#)
 - [Verify CentOS Version](#)
 - [Verify Red Hat Version](#)
- [Partition Size](#)
- [Information about the Example Site](#)

Required Version

 SCS requires either RHEL or CentOS version 7.9 or later are installed. Prior versions are **not** supported.

Check the Linux release version to verify the latest update is running before proceeding.

Verify CentOS Version

To verify the version of CentOS installed:

```
> cat /etc/centos-release
```

Verify Red Hat Version

To verify the version of RHEL installed:

```
> cat /etc/redhat-release
```

Partition Size

Following is the set of requirements for partitioning specific to the OS capacity.

Device	Mount	Partition type	File system type	Size	Options
/dev/sda1	/boot	GPT	ext4	10 GiB	Bootable
/dev/sda2	/	GPT	ext4	20 GiB	NA
/dev/sda3	/var	GPT	ext4	>=120GiB (expand to fill disk)	NA

Information about the Example Site

The table shown below contains information about the example site being set up:

Username	root		
Password	datacore		
Hostname	scs-lab1.datacore.internal		
Application Network	IP Address	172.16.33.10/24	

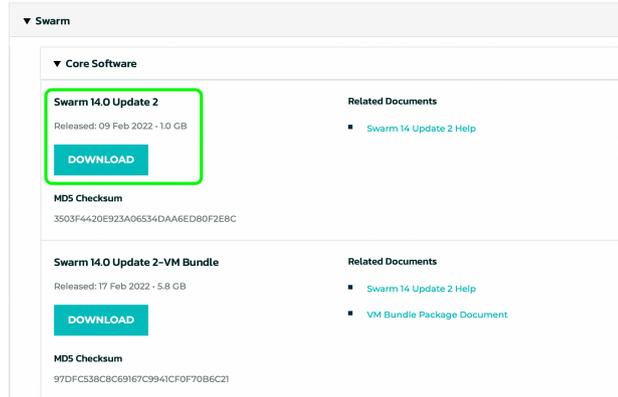
Private Network	IP Address	192.168.9.10/24
Private (Storage) Subnet	IP Address	192.168.9.0/24
Network Gateway	172.16.33.1	
NTP	172.16.33.4,172.16.33.5	

✔ Next, [download Swarm bundle 14.0 package.](#)

Swarm 14.0 Bundle Package

Refer to the following steps to download the Swarm 14.0 bundle package required for SCS installation and setup:

1. Log in to <https://datacore.custhelp.com/app/downloads/downloads>.
2. Download **Swarm 14.x** package (e.g., Swarm-v14.0-Update2-20220209.zip).



3. Unzip Swarm 14 package to /root/datacore.

```
[root@scs ~]# ll datacore/Swarm-v14.0
total 20
d-wxrw--wt. 2 root root 4096 Jan 8 17:07 Elasticsearch
d-wxrw--wt. 2 root root 4096 Jan 8 17:07 Gateway
d-wxrw--wt. 2 root root 4096 Jan 8 17:06 Platform
-rwxr-xr-x. 1 root root 2376 Jan 8 17:05 README.txt
d-wxrw--wt. 2 root root 4096 Jan 8 17:05 UI
```

✔ Next, [install the SCS RPM](#).

SCS Installation

- [Online SCS Installation](#)
- [Offline SCS Installation](#)

Online SCS Installation

Refer to the following steps to install SCS RPM with Internet access:

1. Install the EPEL release.

- a. Download the latest EPEL package `curl -O https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm`
- b. Install the downloaded RPM `yum install epel-release-latest-7.noarch.rpm`

2. If running Red Hat, enable the server extras repository:

```
subscription-manager repos --enable=rhel-7-server-extras-rpms
```

3. Download Kubic Repo for Podman 3.

Note: Though this repo points to Suse, it is the official location podman has chosen to publish the RPM for RHEL/CentOS 7.

```
[root@scs-rhel ~]# curl -L "https://download.opensuse.org/repositories/devel:kubic:libcontai:
% Total      % Received % Xferd      Average Speed      Time      Time      Time      Current
           Dload  Upload    Total     Spent     Left     Speed
100  403  100    403    0    0    655      0  --:--:--  --:--:--  --:--:--   655
100  404  100    404    0    0    549      0  --:--:--  --:--:--  --:--:--   549
100  405  100    405    0    0    473      0  --:--:--  --:--:--  --:--:--   473
100  359  100    359    0    0    366      0  --:--:--  --:--:--  --:~:~:~   366
[root@scs-rhel ~]# ls -l /etc/yum.repos.d/
total 340
drwxr-xr-x.  2 root root   4096 Mar 17 14:33 ./
drwxr-xr-x. 77 root root   4096 Mar 17 13:24 ../
-rw-r--r--.  1 root root    359 Mar 17 14:33 devel:kubic:libcontainers:stable.repo
-rw-r--r--.  1 root root   1358 Sep  4 2021 epel.repo
-rw-r--r--.  1 root root   1457 Sep  4 2021 epel-testing.repo
-rw-r--r--.  1 root root 325697 Mar 17 13:08 redhat.repo
[root@scs-rhel ~]# cat /etc/yum.repos.d/devel\kubic\libcontainers\stable.repo
[devel_kubic_libcontainers_stable]
name=Stable Releases of Upstream github.com/containers packages (CentOS_7)
type=rpm-md
baseurl=https://download.opensuse.org/repositories/devel:/kubic:/libcontainers:/stable/CentO
gpgcheck=1
gpgkey=https://download.opensuse.org/repositories/devel:/kubic:/libcontainers:/stable/CentO:
enabled=1
[root@scs-rhel ~]#
```

4. List the enabled repos using `yum repolist enabled` to verify Kubic is on the list.

5. Install SCS build `yum -y install swarm-scs-1.1.0-1.x86_64.rpm`.

6. Check the `scsctl help` command to verify the SCS is installed.

```
usage: scsctl [-v | --info-log | --debug-log | --trace-log]
             [--user USER | --token TOKEN]
             options: ...
```

Provides basic control and visibility into the Platform service.

optional arguments:

```
--user USER      User name and (optionally) password.
                  If password is included, then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN     Authentication token. (default: None)
```

output options:

```
These options affect the way output is displayed.
-v, --verbose     Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always set.
--info-log        Set info-level verbosity (equivalent to -v). (default: None)
--debug-log       Set debug-level verbosity (equivalent to -vv). (default: None)
--trace-log       Set trace-level verbosity (equivalent to -vvv). (default: None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example:

options:

```
auth             Manages persisted credentials for contacting the Platform API.
backup           Manages backing up and restoring the Platform service.
diagnostics      Performs diagnostics on the Platform server setup.
init             Utilities for setting up a Swarm environment.
license          Manages the license for this Swarm ecosystem installation.
repo            Manages the Platform repository. The repository contains installed components.
system           Performs administrative operations on the system.
help            Outputs this help and exits.
```

```
ERROR: Cannot connect to the Platform API service at 172.16.33.10:8095.
Please ensure the service is up and running.
```

✔ Next, [run the initialization wizard](#).

Offline SCS Installation

- [Minimal Requirements and Needs](#)

Minimal Requirements and Needs

Requirement	Need
CentOS/RHEL	Minimum 7.9 version or higher. See Setup RHEL/CentOS for SCS .
Offline repository	Creating an offline repository of RPM dependencies is required to install SCS on VMs with air-gapped network.
io_pause container image	k8s.gcr.io_pause:3.2 is a container and the offline copy is required to hold the network namespace for the Podman.

There are two ways to set up repositories:

- Follow the instructions to manually create a local repo then use it to install SCS offline.
- Download the repo.tar and use it to install SCS offline.

Contact DataCore Support team to download repo.tar for installing SCS offline.

- [Manually Create a Local Repo](#)
- [Install SCS Offline](#)

Manually Create a Local Repo

- [External Libraries for SCS](#)
- [Set up the Environment](#)
- [SCS Dependencies](#)
 - [Steps to Check SCS Dependencies](#)
 - [Download SCS Dependencies](#)
 - [Download Podman 3.0.1 Repo](#)
- [Create a Local Repo](#)
 - [Create an offline Pause Container](#)

i Manually creating a local repo requires *initial* access to the Internet to retrieve all dependencies, then the machine can be air-gapped.

External Libraries for SCS

Three external libraries to install SCS are:

- Podman 3.0.1 - Required for HTTP support. Version 3.0.1 is the source from the Kubic repo maintained by SUSE.
- Pause Container - The `k8s.gcr.io/pause:3.2` is a container to hold the network namespace for the pod.
- Bash Autocompletion - Required for tabbing on `scsctl`. Sourced from the EPEL repository for CentOS.

Set up the Environment

Offline installation for CentOS requires RPMs and dependencies for all those packages on a repository. This is challenging in air-gapped networks or networks with no Internet access so the best way is to create a local offline repository.

Minimum installation requirements are:

1. Install a VM with CentOS 7.9 (or the previous version) and a “Minimal Server” profile.
2. Run `yum` to resolve dependencies for a package.
3. CentOS 7 server with Internet access and some disk space to pull the repo files.

SCS Dependencies

Install a set of packages to download RPMs and dependencies then create a repository. RPMs and dependencies use the `downloadonly` yum-plugin to pull. Use a tool “`createrepo`” to create the repository for offline usage.

Steps to Check SCS Dependencies

1. Install the yum-downloadonly plugin.

```
yum -y install yum-plugin-downloadonly
```

By default, this package comes in large installations.

2. Create a directory for download packages.

```
mkdir packages
```

3. Query the RPM dependencies.

```
rpm -qpR swarm-scs-{version}.x86_64.rpm
```

Result:

```
glibc >= 2.17
bash-completion
bash-completion-extras
dhcp
python3
iproute
chrony
NetworkManager
rsyslog
logrotate
cronie
/bin/sh
/bin/sh
/bin/sh
rpmlib(PayloadFilesHavePrefix) <= 4.0-1
rpmlib(CompressedFileNames) <= 3.0.4-1
```

Download SCS Dependencies

Pull the following from the above list to download SCS and RPM's dependencies:

Query	Command
bash-completion	yum -y install --downloadonly --downloadaddir=/root/packages/ bash-completion
bash-completion-extras	yum -y install --downloadonly --downloadaddir=/root/packages/ bash-completion-extras
chrony	yum -y install --downloadonly --downloadaddir=/root/packages/ chrony
cronie	yum -y install --downloadonly --downloadaddir=/root/packages/ cronie
dhcp	yum -y install --downloadonly --downloadaddir=/root/packages/ dhcp
iproute	yum -y install --downloadonly --downloadaddir=/root/packages/ iproute
logrotate	yum -y install --downloadonly --downloadaddir=/root/packages/ logrotate
NetworkManager	yum -y install --downloadonly --downloadaddir=/root/packages/ NetworkManager
python3	yum -y install --downloadonly --downloadaddir=/root/packages/ python3
rsyslog	yum -y install --downloadonly --downloadaddir=/root/packages/ rsyslog

Download Podman 3.0.1 Repo

To pull the correct version and its dependencies, enable the Kubic repository.

1. Download Kubic repo from SUSE.

```
cd /etc/yum.repo.d/
curl -L -o /etc/yum.repos.d/devel:kubic:libcontainers:stable.repo
https://download.opensuse.org/repositories/devel:kubic:libcontainers:stable/CentOS_7/devel:}
```

2. Download Podman packages and dependencies.

```
yum -y install --downloadonly --downloaddir=/root/packages/ podman
```

Result:

```
yum -y install --downloadonly --downloaddir=/root/packages/ podman
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
base:
epel:
extras:
updates: mirror.sjc02.svwh.net
Resolving Dependencies
--> Running transaction check
--> Package podman.x86_64 0:3.0.1-2.el7.3.1 will be installed
--> Processing Dependency: podman-plugins = 3.0.1-2.el7.3.1 for package: podman-3.0.1-2.el7.3.1.x
--> Processing Dependency: slirp4netns >= 0.3.0-2 for package: podman-3.0.1-2.el7.3.1.x86_64
--> Processing Dependency: runc >= 2:1.0.0-148.rc93 for package: podman-3.0.1-2.el7.3.1.x86_64
--> Processing Dependency: containers-common >= 4:1-4 for package: podman-3.0.1-2.el7.3.1.x86_64
--> Processing Dependency: containernetworking-plugins >= 0.9.1-1 for package: podman-3.0.1-2.el7
--> Processing Dependency: common >= 2:2.0.16-1 for package: podman-3.0.1-2.el7.3.1.x86_64
--> Processing Dependency: nftables for package: podman-3.0.1-2.el7.3.1.x86_64
--> Processing Dependency: container-selinux for package: podman-3.0.1-2.el7.3.1.x86_64
--> Processing Dependency: catatonit for package: podman-3.0.1-2.el7.3.1.x86_64
--> Running transaction check
--> Package catatonit.x86_64 0:0.1.5-6.el7.3.1 will be installed
--> Package common.x86_64 2:2.0.29-1.el7.3.2 will be installed
--> Package container-selinux.noarch 2:2.119.2-1.911c772.el7_8 will be installed
--> Processing Dependency: policycoreutils-python for package: 2:container-selinux-2.119.2-1.911c
--> Package containernetworking-plugins.x86_64 0:1.0.0-0.2.rc1.el7.6.2 will be installed
--> Package containers-common.noarch 4:1-18.el7.20.1 will be installed
--> Processing Dependency: fuse-overlayfs for package: 4:containers-common-1-18.el7.20.1.noarch
--> Package nftables.x86_64 1:0.8-14.el7 will be installed
--> Processing Dependency: libnftnl.so.7(LIBNFTNL_5)(64bit) for package: 1:nftables-0.8-14.el7.x8
--> Processing Dependency: libnftnl.so.7()(64bit) for package: 1:nftables-0.8-14.el7.x86_64
--> Package podman-plugins.x86_64 0:3.0.1-2.el7.3.1 will be installed
--> Processing Dependency: dnsmasq for package: podman-plugins-3.0.1-2.el7.3.1.x86_64
--> Package runc.x86_64 2:1.0.1-0.el7.1.2 will be installed
--> Processing Dependency: criu for package: 2:runc-1.0.1-0.el7.1.2.x86_64
--> Package slirp4netns.x86_64 0:1.1.8-4.el7.7.1 will be installed
--> Processing Dependency: libslirp.so.0(SLIRP_4.1)(64bit) for package: slirp4netns-1.1.8-4.el7.7
--> Processing Dependency: libslirp.so.0(SLIRP_4.0)(64bit) for package: slirp4netns-1.1.8-4.el7.7
--> Processing Dependency: libslirp.so.0()(64bit) for package: slirp4netns-1.1.8-4.el7.7.1.x86_64
--> Running transaction check
--> Package criu.x86_64 0:3.12-2.el7 will be installed
--> Processing Dependency: libprotobuf-c.so.1(LIBPROTOBUF_C_1.0.0)(64bit) for package: criu-3.12-
--> Processing Dependency: libprotobuf-c.so.1()(64bit) for package: criu-3.12-2.el7.x86_64
--> Processing Dependency: libnet.so.1()(64bit) for package: criu-3.12-2.el7.x86_64
--> Package dnsmasq.x86_64 0:2.76-17.el7_9.3 will be installed
--> Processing Dependency: libnettle.so.4()(64bit) for package: dnsmasq-2.76-17.el7_9.3.x86_64
--> Package fuse-overlayfs.x86_64 0:1.5.0-1.el7.1.1 will be installed
--> Processing Dependency: libfuse3.so.3(FUSE_3.2)(64bit) for package: fuse-overlayfs-1.5.0-1.el7
--> Processing Dependency: libfuse3.so.3(FUSE_3.0)(64bit) for package: fuse-overlayfs-1.5.0-1.el7
--> Processing Dependency: fuse3 for package: fuse-overlayfs-1.5.0-1.el7.1.1.x86_64
--> Processing Dependency: libfuse3.so.3()(64bit) for package: fuse-overlayfs-1.5.0-1.el7.1.1.x86
--> Package libnftnl.x86_64 0:1.0.8-3.el7 will be installed
--> Package libslirp.x86_64 0:4.3.1-4.el7 will be installed
--> Package policycoreutils-python.x86_64 0:2.5-34.el7 will be installed
--> Processing Dependency: setools-libs >= 3.3.8-4 for package: policycoreutils-python-2.5-34.el7
--> Processing Dependency: libsemanage-python >= 2.5-14 for package: policycoreutils-python-2.5-3
```

```
--> Processing Dependency: audit-libs-python >= 2.1.3-4 for package: polycoreutils-python-2.5-3
--> Processing Dependency: python-IPy for package: polycoreutils-python-2.5-34.el7.x86_64
--> Processing Dependency: libqpol.so.1(VERS_1.4)(64bit) for package: polycoreutils-python-2.5-
--> Processing Dependency: libqpol.so.1(VERS_1.2)(64bit) for package: polycoreutils-python-2.5-
--> Processing Dependency: libcgroup for package: polycoreutils-python-2.5-34.el7.x86_64
--> Processing Dependency: libapol.so.4(VERS_4.0)(64bit) for package: polycoreutils-python-2.5-
--> Processing Dependency: checkpolicy for package: polycoreutils-python-2.5-34.el7.x86_64
--> Processing Dependency: libqpol.so.1()(64bit) for package: polycoreutils-python-2.5-34.el7.x
--> Processing Dependency: libapol.so.4()(64bit) for package: polycoreutils-python-2.5-34.el7.x
--> Running transaction check
---> Package audit-libs-python.x86_64 0:2.8.5-4.el7 will be installed
---> Package checkpolicy.x86_64 0:2.5-8.el7 will be installed
---> Package fuse3.x86_64 0:3.6.1-4.el7 will be installed
--> Processing Dependency: /etc/fuse.conf for package: fuse3-3.6.1-4.el7.x86_64
---> Package fuse3-libs.x86_64 0:3.6.1-4.el7 will be installed
---> Package libcgroup.x86_64 0:0.41-21.el7 will be installed
---> Package libnet.x86_64 0:1.1.6-7.el7 will be installed
---> Package libsemanage-python.x86_64 0:2.5-14.el7 will be installed
---> Package nettle.x86_64 0:2.7.1-9.el7_9 will be installed
---> Package protobuf-c.x86_64 0:1.0.2-3.el7 will be installed
---> Package python-IPy.noarch 0:0.75-6.el7 will be installed
---> Package setools-libs.x86_64 0:3.3.8-4.el7 will be installed
--> Running transaction check
---> Package fuse.x86_64 0:2.9.2-11.el7 will be installed
--> Finished Dependency Resolution
```

Dependencies Resolved

```
=====
```

Package	Arch	Version	Repository	Size
=====				
Installing:				
podman	x86_64	3.0.1-2.el7.3.1	kubic	21 M
Installing for dependencies:				
audit-libs-python	x86_64	2.8.5-4.el7	base	76 k
catatonit	x86_64	0.1.5-6.el7.3.1	kubic	326 k
checkpolicy	x86_64	2.5-8.el7	base	295 k
common	x86_64	2:2.0.29-1.el7.3.2	kubic	45 k
container-selinux	noarch	2:2.119.2-1.911c772.el7_8	extras	40 k
containernetworking-plugins	x86_64	1.0.0-0.2.rc1.el7.6.2	kubic	39 M
containers-common	noarch	4:1-18.el7.20.1	kubic	57 k
criu	x86_64	3.12-2.el7	base	453 k
dnsmasq	x86_64	2.76-17.el7_9.3	updates	281 k
fuse	x86_64	2.9.2-11.el7	base	86 k
fuse-overlayfs	x86_64	1.5.0-1.el7.1.1	kubic	67 k
fuse3	x86_64	3.6.1-4.el7	extras	47 k
fuse3-libs	x86_64	3.6.1-4.el7	extras	82 k
libcgroup	x86_64	0.41-21.el7	base	66 k
libnet	x86_64	1.1.6-7.el7	base	59 k
libnftnl	x86_64	1.0.8-3.el7	base	78 k
libsemanage-python	x86_64	2.5-14.el7	base	113 k
libslirp	x86_64	4.3.1-4.el7	kubic	60 k
nettle	x86_64	2.7.1-9.el7_9	updates	328 k
nftables	x86_64	1:0.8-14.el7	base	186 k
podman-plugins	x86_64	3.0.1-2.el7.3.1	kubic	2.5 M
polycoreutils-python	x86_64	2.5-34.el7	base	457 k
protobuf-c	x86_64	1.0.2-3.el7	base	28 k
python-IPy	noarch	0.75-6.el7	base	32 k
runc	x86_64	2:1.0.1-0.el7.1.2	kubic	6.0 M
setools-libs	x86_64	3.3.8-4.el7	base	620 k
slirp4netns	x86_64	1.1.8-4.el7.7.1	kubic	49 k

Transaction Summary

```

=====
Install 1 Package (+27 Dependent packages)

Total download size: 72 M
Installed size: 180 M
Background downloading packages, then exiting:
(1/28): container-selinux-2.119.2-1.911c772.el7_8.noarch.rpm | 40 kB 00:00:00
(2/28): audit-libs-python-2.8.5-4.el7.x86_64.rpm | 76 kB 00:00:00
(3/28): checkpolicy-2.5-8.el7.x86_64.rpm | 295 kB 00:00:00
warning: /root/packages/common-2.0.29-1.el7.3.2.x86_64.rpm: Header V3 RSA/SHA256 Signature, key I
Public key for common-2.0.29-1.el7.3.2.x86_64.rpm is not installed
(4/28): common-2.0.29-1.el7.3.2.x86_64.rpm | 45 kB 00:00:01
(5/28): catatonit-0.1.5-6.el7.3.1.x86_64.rpm | 326 kB 00:00:01
(6/28): fuse-2.9.2-11.el7.x86_64.rpm | 86 kB 00:00:00
(7/28): containers-common-1-18.el7.20.1.noarch.rpm | 57 kB 00:00:00
(8/28): dnsmasq-2.76-17.el7_9.3.x86_64.rpm | 281 kB 00:00:00
(9/28): fuse3-3.6.1-4.el7.x86_64.rpm | 47 kB 00:00:00
(10/28): fuse-overlayfs-1.5.0-1.el7.1.1.x86_64.rpm | 67 kB 00:00:00
(11/28): libcgrouop-0.41-21.el7.x86_64.rpm | 66 kB 00:00:00
(12/28): fuse3-libs-3.6.1-4.el7.x86_64.rpm | 82 kB 00:00:00
(13/28): libnet-1.1.6-7.el7.x86_64.rpm | 59 kB 00:00:00
(14/28): libnftnl-1.0.8-3.el7.x86_64.rpm | 78 kB 00:00:00
(15/28): libsemanage-python-2.5-14.el7.x86_64.rpm | 113 kB 00:00:00
(16/28): libslirp-4.3.1-4.el7.x86_64.rpm | 60 kB 00:00:00
(17/28): nettle-2.7.1-9.el7_9.x86_64.rpm | 328 kB 00:00:00
(18/28): nftables-0.8-14.el7.x86_64.rpm | 186 kB 00:00:00
(19/28): criu-3.12-2.el7.x86_64.rpm | 453 kB 00:00:02
(20/28): podman-3.0.1-2.el7.3.1.x86_64.rpm | 21 MB 00:00:07
(21/28): protobuf-c-1.0.2-3.el7.x86_64.rpm | 28 kB 00:00:00
(22/28): polycycoreutils-python-2.5-34.el7.x86_64.rpm | 457 kB 00:00:00
(23/28): python-IPy-0.75-6.el7.noarch.rpm | 32 kB 00:00:00
(24/28): podman-plugins-3.0.1-2.el7.3.1.x86_64.rpm | 2.5 MB 00:00:02
(25/28): setools-libs-3.3.8-4.el7.x86_64.rpm | 620 kB 00:00:00
(26/28): runc-1.0.1-0.el7.1.2.x86_64.rpm | 6.0 MB 00:00:03
(27/28): slirp4netns-1.1.8-4.el7.7.1.x86_64.rpm | 49 kB 00:00:00
(28/28): containernetworking-plugins-1.0.0-0.2.rc1.el7.6.2.x86 | 39 MB 00:00:17
-----
Total 3.8 MB/s | 72 MB 00:18
exiting because "Download Only" specified

```

Note

Downloaded packages are not enough to install SCS server due to third-party external packages. Suggest query on each downloaded RPM packages to get the new list for additional download.

Contact DataCore Support if facing any issue and need assistance for manually creating a local repo.

Create a Local Repo

1. Run `createrepo` against the packages folder.

```

[root@dhcp-128 ~]# createrepo packages/
Spawning worker 0 with 18 pkgs
Spawning worker 1 with 18 pkgs
Workers Finished
Saving Primary metadata
Saving file lists metadata
Saving other metadata
Generating sqlite DBs
Sqlite DBs complete

```

2. Tar up the directory and its packages.

```
cd /root/
tar -cvf repo.tar packages/
```

Create an offline Pause Container

1. Install Podman on CentOS 7.

```
yum -y install podman
```

2. Pull the Pause Container.

3. Create a new pod to automatically pull the *k8s.gcr.io/pause:3.2* image.

```
podman pod create
fc299d05422a55e46774aa06a66bc853109aeb4b1d7078f81f9ffa4eef26c8e0
```

4. List the pulled images.

```
podman images
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
k8s.gcr.io/pause    3.2         80d28bedfe5d     22 months ago   688 kB
```

Note:

Tag *k8s.gcr.io/pause* with 3.2 or higher.

5. Export Pause Container and save it for offline installation.

```
podman save -o k8s.gcr.io_pause_3.2.tar k8s.gcr.io/pause:3.2
```

```
ll
total 680
-rw-r--r--. 1 root root 692224 Dec  7 20:10 k8s.gcr.io_pause_3.2.tar
```

✔ Next, see [Install SCS Offline](#).

Install SCS Offline

Copy the *local repo* (repo.tar) and *Pause Container* (k8s.gcr.io_pause_3.2.tar) to the CentOS 7 to install SCS Server 14.

1. Create a new working directory for SCS Installation.

```
mkdir /root/datacore
```

2. Copy **repo.tar** and **k8s.gcr.io_pause_3.2.tar** to `"/root/datacore/"`.
3. Create a local repo - `"/etc/yum.repos.d/local.repo"`.

```
[local]
name=Everything we need to install scs
baseurl=file:///root/datacore/packages
enabled=true
gpgcheck=0
```

4. Create a local repo for installing Podman. Example: `"/etc/yum.repos.d/devel:kubic:libcontainers:stable.repo"`.

Note:

Verify the file name matches exactly.

```
[kubic]
name=Stable Releases of Upstream packages (CentOS_7)
baseurl=file:///root/datacore/packages
enabled=true
gpgcheck=0
```

5. Disable *base*, *updates*, *extras*, and *CentOsPlus* repositories. Add `enable=0` to `[base]`, `[updates]`, `[extras]` and `[centosplus]` session of `"/etc/yum.repos.d/CentOS-Base.repo"` or rename it.

```
[base]
name=CentOS-$releasever - Base
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=os&infra=$:
#baseurl=http://mirror.centos.org/centos/$releasever/os/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
enabled=0

#released updates
[updates]
name=CentOS-$releasever - Updates
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=updates&inf
#baseurl=http://mirror.centos.org/centos/$releasever/updates/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
enabled=0

#additional packages that may be useful
[extras]
name=CentOS-$releasever - Extras
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=extras&inf
#baseurl=http://mirror.centos.org/centos/$releasever/extras/$basearch/
gpgcheck=1
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7

#additional packages that extend functionality of existing packages
[centosplus]
name=CentOS-$releasever - Plus
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=centosplus&inf
#baseurl=http://mirror.centos.org/centos/$releasever/centosplus/$basearch/
gpgcheck=1
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
```

6. Flush repo metadata.

```
yum clean all
yum repolist
```

7. Install an NTP Server. Do not use Active Directory as an NTP Server Source.

```
yum -y install chrony
```

8. Configure sync with local NTP Server.

IMPORTANT:

NTP does not sync with the Active Directory.

9. Install Podman 3.0.1.

```
yum -y install podman
```

10. Verify the Podman 3.0.1 installation is successful.

```
podman -v
podman version 3.0.1
```

11. Load Pause Container.

```
podman image load -i /root/datacore/k8s.gcr.io_pause_3.2.tar
```

12. Verify the Pause Container is loaded and the container TAG label is 3.2.

```
podman images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
k8s.gcr.io/pause	3.2	80d28bedfe5d	22 months ago	688 kB

13. Refer to [Online SCS Installation](#) and follow step 5 onwards.

Run the Initialization Wizard

Run the initialization wizard once the installation is complete:

Important

- CentOS/RHEL 7.9 of SCS must have access to Internet.
- Use `--debug-log` flag to see debug logging on the screen:
`scsctl init wizard --allow-all-system-updates --debug-log`

Refer to the following steps:

1. Run SCS init wizard with default settings `scsctl init wizard --allow-all-system-updates`.
2. Provide a Swarm site name which is a human-oriented name used to identify the particular Swarm site within the overall organization. The `scs-lab1.datacore.internal` example used below generates other default names later. A DNS-style hostname format is recommended (where the value **does not** need to be DNS resolvable).

```
[root@zorc swarm_v14-0]# scsctl init wizard --allow-all-system-updates
Running step [1/33]: Set site name
Please enter a name for this Swarm site. It should be unique within your organization:
scs-lab1.datacore.internal
```

3. Select the internal network interface (see [Network Planning](#) for detail); this interface contains the private subnet restricted to Swarm use.

```
Running step [2/33]: Choose Swarm-internal interface
Please specify the network interface that will be used for internal Swarm operations:
lo
ens192
> ens224
preview
3: ens224: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen
link/ether 00:50:56:83:3a:1e brd ff:ff:ff:ff:ff:ff
inet 192.168.9.10/24 brd 192.168.9.255 scope global noprefixroute ens224
```

4. Ensure the Swarm internal interface is correctly defined and running. Enter 'Y' to verify the internal network IP address and network mask if the network interface has been configured otherwise enter 'N' and provide the interface IP address and subnet mask when prompted.

```
Running step [3/33]: Ensure NetworkManager is up
Running step [4/33]: Detect external interface
Running step [5/33]: Detect single/dual network mode
Running step [6/33]: Define Swarm-internal network
The internal interface requires a *static* IP address to be defined on it!
It looks like your internal interface is already configured with an IP address: 192.168.9.10

Do you wish to continue to use this address and netmask? [Y/n] :
Y
```

5. SCS wizard verifies the network interface definition matches the configured values.

```
Running step [7/33]: Ensure Swarm-internal network is defined
An existing network definition for interface ens224 has been found. It will be overwritten.
Running step [8/33]: Ensure Swarm-internal network is up
```

6. The wizard performs the following actions:
 - a. Generates internal SCS component references.

```
Running step [9/33]: Generate platform group name
Running step [10/33]: Generate network boot group name
```

b. Verifies podman is up-to-date and available for use.

```
Running step [11/33]: Ensure up-to-date version of podman (may take a while)
Running step [12/33]: Enable API access to podman
Running step [13/33]: Establish connection to podman API
```

c. Starts loading SCS containers to podman into the runtime environment.

```
Running step [14/33]: Load Platform container images (may take a while)
  Loading image: quay.io_coreos_etcd-v3.3
  Loading image: swarm-platform-pxelinux-14.1.0
  Loading image: swarm-platform-full-14.1.0
  Loading image: swarm-platform-tftp-14.1.0
Running step [15/33]: Make networking survive firewall restarts
Running step [16/33]: Prepare etcd data volume
Running step [17/33]: Prepare API repo volume
```

d. Sets up the automatic launch of SCS services on system startup.

```
Running step [18/33]: Build pod definition for Platform services
Running step [19/33]: Enable systemd management of Platform services
Running step [20/33]: Launch the API service (may take a while)
```

e. Secures the SCS.

```
Running step [21/33]: Lock down network access to Swarm Platform
Running step [22/33]: Enable TFTP via podman bridge
Running step [23/33]: Enable NAT from internal network
```

f. Performs the initial configuration of SCS services.

```
Running step [24/33]: Enable syslog from internal network
Running step [25/33]: Create global platform group
Running step [26/33]: Set known default Platform settings
Running step [27/33]: Create network booting group
Running step [28/33]: Pre-register network booting instance
Running step [29/33]: Configure NTP server
Running step [30/33]: Configure syslog server
```

g. Shows “next steps” to customize the installation of site-specific needs.

```
Running step [33/33]: Show the next steps after this wizard
*****
Congratulations, your Swarm Platform is now initialized!
Please use the following command to learn about adding Swarm storage software:
  > scsctl repo component add help

Once Swarm storage has been added, please run the following to complete your install
  > scsctl diagnostics config scan_missing
  > scsctl init dhcp

To learn about installing a Swarm license:
  > scsctl license add help

If you need this list again, please run:
  > scsctl init wizard --next-steps
*****
```

7. The basic setup of the SCS is complete. Continue by adding the Swarm Storage components as follows.

Note: It is recommended to avoid “Encryption at Rest” unless specified by DataCore Support team. Contact DataCore Support team for configurations after completing the above steps if “Encryption at Rest” use is desired. Choose “Later at the group and/or instance level” to skip this step.

```
scsctl repo component add -f swarm-scs-storage-14.0.1.tgz
Missing setting: storage/disk.encryptNewVolumes
```

```
How would you like to update this setting?
> Now as a component-wide default
  Later at the group and/or instance level
```

8. The "disk.volumes" setting tells Swarm Storage which storage devices to use. Enter "all" and press *Enter*.

```
disk.volumes [type: str] (Required. Specifies the volume storage devices for Swarm to use. \
  all
added: storage - 14.0.1 (14.0.1) (113.35 MB sent)... ..
```

9. Enter a group name in the FQDN format e.g., `objstor.scs-lab1.datacore.internal`. This value is used as the cluster name.

```
Please enter a name for the group (FQDN format encouraged) - Required to be able to boot in:
  objstor.scs-lab1.datacore.internal
```

10. Enter a description for the group e.g., `DataCore Swarm Object Storage`.

```
Please enter a description for the group (purpose, etc). [OPTIONAL]:
  DataCore Swarm Object Storage
group added
```

```
=====
Please run the following command to check for missing config in the new group:
  > scsctl diagnostics config scan_missing
=====
```

✔ Next, [finalize Swarm configuration settings](#).

Add the Swarm Storage Component

Add the storage software bundle for the cluster.

```
scsctl repo component add -f {storage component tarball file}
```

Enter `all` when prompted for the `disk.values` setting default value unless otherwise instructed by DataCore. The CLI prompts it again during the [finalization of Swarm configuration settings](#) if this setting is skipped.

Add a group when prompted to define a cluster for storage nodes.

 The group name is used as the cluster name. The group description is optional and for human administrator purposes.

Example:

```
[root@zorcl ~]# scsctl repo component add -f swarm-scs-storage-14.0.1.tgz
Missing setting: storage/disk.volumes
```

The following setting(s) require a default value:

```
disk.volumes [type: str] (Required. Specifies the volume storage devices for Swarm to use. Valid all
```

```
added: storage - 14.0.1 (14.0.1)
```

```
Please enter a name for the group (FQDN format encouraged) - Required to be able to boot instance
a.storage.cluster
```

```
Please enter a description for the group (purpose, etc). [OPTIONAL]:
Store it all!
```

```
group added
```

```
=====
Please run the following command to check for missing config in the new group:
> scsctl diagnostics config scan_missing
=====
[root@zorcl ~]#
```

 [Finalize Swarm configuration settings](#) next.

Finalize Swarm Configuration Settings

Some Swarm configuration settings are site-specific and related to:

- **DNS settings** - These are required for [DHCP configuration](#).
- **Administrative username and password** - SCS does not accept any unauthenticated commands once the administrator username and password are set.
- **Disk encryption settings** - These settings are related to "Encryption at Rest" and should be skipped unless specified by DataCore Support.

Info

This process is interrupted to allow `scsctl` to be configured with the administrative username and password for future interactions with SCS once the administrative username and password are set. To perform this configuration run:

```
scsctl auth login --user [administrative user name]
```

This command securely prompts the administrative password and authenticates to SCS. Re-run the configuration scan command to resume once the CLI is logged in:

```
scsctl diagnostics config scan_missing
```

Finalize configuration settings

Note

Input characters such as 'admin.userName', 'admin.password', 'encryptionKeys' are hidden when configuring Swarm settings. Best practice is to copy and paste inputs from a raw text editor to prevent any transcription or transposition errors.

1. Run `scsctl diagnostics config scan_missing`

```
=====
Component: network_boot
-----
Name: network.dnsServers
Type: array[str]
Description: Required. DNS servers to be used.
-----
Component Group: network_boot/192.168.9.0x24.network_boot.scs-lab1.datacore.internal [DEFAULT]
Instances cannot currently inherit these from the group (since they have not been defined at
the group level)

If any new instances are added, please define these settings, or define them at the group level
```

2. Press 'Enter' to apply settings when prompted for the DNS servers.

```
-----
Name: network.dnsServers
Type: array[str]
Description: Required. DNS servers to be used.
-----
Missing setting: network_boot/network.dnsServers

Where would you like to update this setting?
> As a default at group level
   For each instance in the group
   Skip this setting for now
```

3. Provide DNS servers.

```

-----
Name: network.dnsServers
Type: array[str]
Description: Required. DNS servers to be used.
-----
Missing setting: network_boot/network.dnsServers
  network.dnsServers [type: array[str]] (Required. DNS servers to be used.):
172.16.33.4,172.16.33.5

=====
Component: platform
-----
Name: admin.password
Type: str
Description: Administrative user password
-----
Name: admin.userName
Type: str
Description: Administrative user name
-----
Name: logging.syslogHost
Type: str
Description: If defined, then the host that component instances should use for syslog logging
-----
Name: network.dnsDomain
Type: str
Description: Required. The DNS domain name that will be used.
-----
Name: network.ntpServers
Type: array[str]
Description: Required. The IP address(es) of one or more NTP servers. If the nodes cannot ac
-----
Component Group: platform/global.platform.scs-lab1.datacore.internal [DEFAULT GROUP]
Instances cannot currently inherit these from the group (since they have not been defined at

If any new instances are added, please define these settings, or define them at the group le
-----
Name: admin.password
Type: str
Description: Administrative user password
-----
Name: admin.userName
Type: str
Description: Administrative user name
-----
Name: network.dnsDomain
Type: str
Description: Required. The DNS domain name that will be used.
-----
    
```

4. Press 'Enter' to apply the password at the group level when prompted for the administrative password.

```

Missing setting: platform/admin.password

Where would you like to update this setting?
> As a default at group level
  For each instance in the group
  Skip this setting for now
    
```

5. Enter the administrator password (e.g., datacore). The input characters are hidden for the password.

```

Missing setting: platform/admin.password

admin.password [type: str ***SECURE***] (Administrative user password):

Please re-enter to confirm:
    
```

- Press 'Enter' to apply the user name at the group level when prompted for the administrative user name.

```
Missing setting: platform/admin.userName
```

```
Where would you like to update this setting?
```

```
> As a default at group level
  For each instance in the group
  Skip this setting for now
```

- Enter the administrator username (e.g., `dcadmin`). The input characters are hidden for the administrator username.

```
Missing setting: platform/admin.userName
```

```
admin.userName [type: str ***SECURE***] (Administrative user name):
```

```
Please re-enter to confirm:
```

```
Authentication state may have changed in the API!
```

```
Please log in, then re-run this command to resume:
```

```
> scsctl auth login --user "{administrative user name}"
> scsctl diagnostics config scan_missing
```

- At this point the SCS API is locked and all requests must be authenticated. Log in to SCS CLI via `scsctl auth login --user [administrative user name]`. The below example uses the sample administrative user name from above. The input characters are hidden.

```
[root@zorc Platform]# scsctl auth login --user dcadmin
Enter password for user "dcadmin":
```

```
Logged in
```

- Continue finalizing configuration settings via `scsctl diagnostics config scan_missing` command once logged in.

```

=====
Component: network_boot
-----
Name: network.dnsServers
Type: array[str]
Description: Required. DNS servers to be used.
-----

Component Group: network_boot/192.168.9.0x24.network_boot.scs-lab1.datacore.internal [DEFAULT]
all settings defined for group
=====
    
```

```

Component: platform
-----
Name: admin.password
Type: str
Description: Administrative user password
-----
Name: admin.userName
Type: str
Description: Administrative user name
-----
Name: logging.syslogHost
Type: str
Description: If defined, then the host that component instances should use for syslog logging
-----
Name: network.dnsDomain
Type: str
Description: Required. The DNS domain name that will be used.
-----
Name: network.ntpServers
Type: array[str]
Description: Required. The IP address(es) of one or more NTP servers. If the nodes cannot ac
    
```

Component Group: platform/global.platform.scs-lab1.datacore.internal [DEFAULT GROUP]

Instances cannot currently inherit these from the group (since they have not been defined at

If any new instances are added, please define these settings, or define them at the group level

10. Press 'Enter' to apply it at the group level when prompted for the DNS domain.

```

-----
Name: network.dnsDomain
Type: str
Description: Required. The DNS domain name that will be used.
-----
    
```

Missing setting: platform/network.dnsDomain

Where would you like to update this setting?

```

> As a default at group level
  For each instance in the group
  Skip this setting for now
    
```

11. Set the DNS domain.

```
-----
Name: network.dnsDomain
Type: str
Description: Required. The DNS domain name that will be used.
-----
```

Missing setting: platform/network.dnsDomain

```
network.dnsDomain [type: str] (Required. The DNS domain name that will be used.):
datacore.internal
=====
```

Component: storage

```
-----
Name: disk.encryptedKeyPrimary
Type: str
Description: The mnemonic name of the encryption key to use for encrypting new Swarm volumes
-----
```

```
Name: disk.encryptedKeys
Type: dict[str,str]
Description: A comma-separated list of mnemonic name and encryption key pairs, used for access
-----
```

```
Name: support.proxyPassword
Type: str
Description: Proxy authentication password
-----
```

Component Group: storage/objstor.scs-lab1.datacore.internal [DEFAULT GROUP]
 Instances cannot currently inherit these from the group (since they have not been defined at

If any new instances are added, please define these settings, or define them at the group level

```
-----
Name: disk.encryptedKeyPrimary
Type: str
Description: The mnemonic name of the encryption key to use for encrypting new Swarm volumes
-----
```

```
Name: disk.encryptedKeys
Type: dict[str,str]
Description: A comma-separated list of mnemonic name and encryption key pairs, used for access
-----
```

```
Name: support.proxyPassword
Type: str
Description: Proxy authentication password
-----
```

12. Select *Skip this setting for now* and press 'Enter'.

Missing setting: storage/disk.encryptedKeyPrimary

```
Where would you like to update this setting?
  As a default at group level
  For each instance in the group
> Skip this setting for now
```

13. The support proxy password applies if an HTTP proxy is needed for automatic health report submission. Select *As a default at group level now* and press 'Enter'.

Missing setting: storage/support.proxyPassword

```
Where would you like to update this setting?
> As a default at group level
  For each instance in the group
  Skip this setting for now
```

14. Enter Proxy Password (e.g., datacore). The input characters are hidden for the password.

Missing setting: storage/support.proxyPassword

support.proxyPassword [type: str ***SECURE***] (Proxy authentication password):

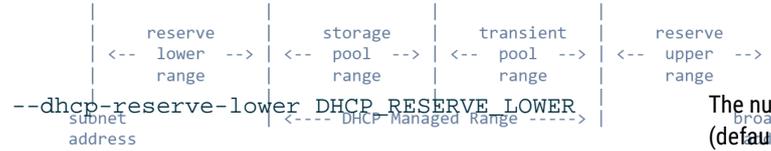
Please re-enter to confirm:

✔ Next, [configure DHCP](#).

Configure DHCP

This `scsctl` initialization step builds a custom configuration file for DHCP restarts the service using the pattern determined in [network planning](#) for the site. Changes in the configuration file specify pools and ranges which DHCP is to manage, as well as other DHCP response data to be used with PXE booting storage nodes.

Subnet Layout:



The Swarm network requires an upper or lower (or both) reserved range defined to configure DHCP. At least one of these parameters must be provided.

<pre>--dhcp-reserve-lower DHCP_RESERVE_LOWER</pre>	<p>The number of IP addresses to reserve in the lower subnet range (default is 0).</p>
<pre>--dhcp-reserve-upper DHCP_RESERVE_UPPER</pre>	<p>The number of IP addresses to reserve in the upper subnet range (default is 0).</p>

For example, set the DHCP reserve lower range.

```
scsctl init dhcp --dhcp-reserve-lower=30
DHCP config updated
```

The configuration values used for DHCP reserve lower or upper range always come from the network planning.

See [additional Swarm configuration](#) for setting up more configuration for DHCP.

✔ Next, [install Swarm license](#).

Additional Swarm Configuration

Enabling `nodeExplorer` is an additional configuration performed after setting up either the lower or upper range or both for DHCP.

To enable the `nodeExporter` for Prometheus, use the below commands.

```
scsctl storage config set -d "metrics.enableNodeExporter=true"  
scsctl storage config set -d "metrics.nodeExporterFrequency=120"
```

Install Swarm License

- [Installing from a file](#)
- [Installing from a URL](#)
- [Confirmation](#)

 Verify the latest Swarm license file is either copied to the SCS server or reachable using HTTP from the SCS server.

Installing from a file

Use `scsctl license add -f {path to license file}` to install the latest Swarm license.

```
scsctl license add -f castor_license.txt
added
```

Installing from a URL

Use `scsctl license add -u {URL of license file}` to install the Swarm license from a URL.

```
scsctl license add -u http://license_server/path/to/license/file
added
```

Confirmation

The Swarm license is ready once `added` appears after the command as shown above. Use `scsctl license show` to visually verify the license is ready.

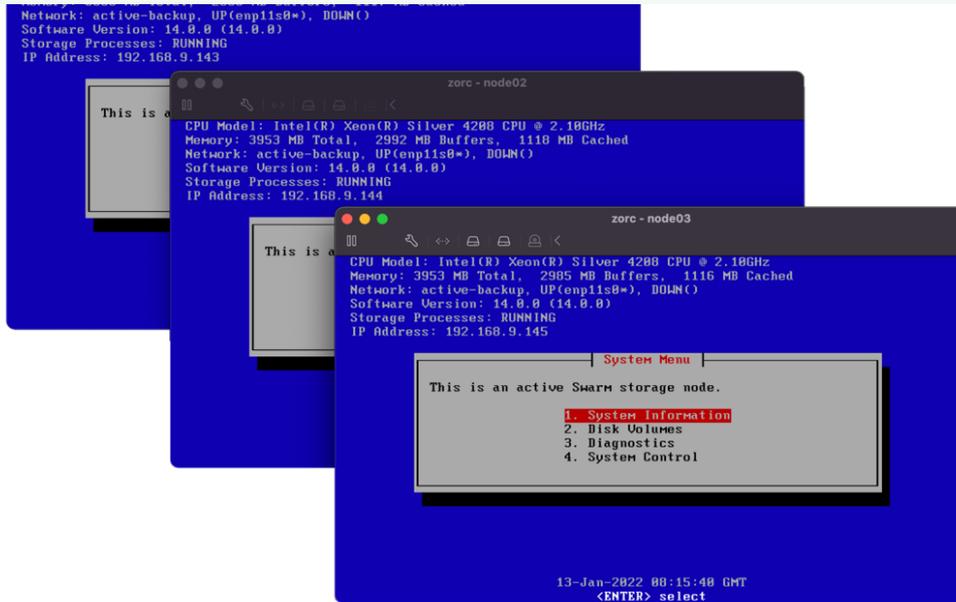
 Next, [power on Swarm virtual storage nodes](#).

Power on the Swarm Storage Nodes

SCS is ready to boot storage nodes after applying the Swarm license. Power on each chassis one by one and SCS assigns an IP address to the storage nodes in sequence. The Swarm storage nodes begin accepting requests after a few minutes.



Next, [setup Elasticsearch cluster](#).



Setup Elasticsearch Cluster

- [Partition Size](#)
- [CentOS setup](#)
- [Download Swarm 14.0 Bundle Package](#)
- [Install Elasticsearch](#)
- [Configure 3 nodes Elasticsearch cluster for Swarm](#)
 - [Create data folder for Swarm Elasticsearch](#)
 - [Configure Elasticsearch Cluster](#)

Partition Size

A 3 nodes Elasticsearch cluster requires 3 virtual machines having the same hardware configurations. Following is the set of requirements for partitioning specific to OS capability:

Device	Mount	Partition type	File system type	Size	Options
/dev/sda1	/boot	GPT	ext4	10 GiB	Bootable
/dev/sda2	/	GPT	ext4	20 GiB	NA
/dev/sda3	/var	GPT	ext4	>=120GiB (<i>expand to Fill Disk</i>)	NA

 It is recommended to run Elasticsearch on SSDs.

CentOS setup

 See [here](#) to setup time synchronization for CentOS 7.

The below table contains information about the example site being set up:

Minimal Install Mode	Hostname	<ul style="list-style-type: none"> • es75-01.datacore.internal • es75-02.datacore.internal • es75-03.datacore.internal
<i>Application Network (Optional)</i>	IP Address	<ul style="list-style-type: none"> • 172.16.33.21/24 • 172.16.33.22/24 • 172.16.33.23/24
<i>Private Network</i>	IP Address	<ul style="list-style-type: none"> • 192.168.9.21/24 • 192.168.9.22/24 • 192.168.9.23/24
Network Gateway		172.16.33.1

NTP	<ul style="list-style-type: none"> • 172.16.33.4 • 172.16.33.5
-----	--

Install the open-vm-tools and run the below commands:

```
yum -y install open-vm-tools
systemctl enable vmttoolsd
systemctl start vmttoolsd
```

Download Swarm 14.0 Bundle Package

See [Swarm 14.0 Bundle Package](#) to download Swarm 14.0 bundle package.

Install Elasticsearch

Refer to the following steps:

1. Update CentOS via `yum -y update` command.
2. Install EPEL `yum -y install epel-release`.
3. Install NTP Server.

```
yum -y install ntp
systemctl enable ntpd
hwclock --systohc
systemctl restart ntpd
```

4. Unzip Swarm 14.0 bundle package to `/root/datacore`.
5. Get the latest Elasticsearch RPM and Swarm Search RPM from the downloaded Swarm bundle package.

```
[root@es75-01 ~]# ll ~/datacore/Elasticsearch/
total 283488
-rwxr-xr-x. 1 root root      38380 Jan  8 18:33 caringo-elasticsearch-search-7.0.3-1.noarch.i
-rwxr-xr-x. 1 root root 290237676 Jan  8 18:33 elasticsearch-7.5.2-x86_64.rpm
-rwxr-xr-x. 1 root root      345 Jan  8 18:33 README.txt
-rwxr-xr-x. 1 root root      1927 Jan  8 18:33 RPM-GPG-KEY
[root@es75-01 ~]#
```

6. Install Swarm RPM public key `rpm --import RPM-GPG-KEY` included with the distribution bundle.
7. Install and configure the Elasticsearch components on each Elasticsearch node.

```
yum -y install elasticsearch-7.5.2-x86_64.rpm
yum -y install caringo-elasticsearch-search-7.0.3-1.noarch.rpm
```

8. Start the Elasticsearch service to verify if installation is finished.

```
systemctl start elasticsearch
systemctl status elasticsearch
```

The Elasticsearch service automatically creates a user and group:

- a. User - elasticsearch
- b. Group - elasticsearch

```
[root@es75-01 ~]# systemctl status elasticsearch
elasticsearch.service - Elasticsearch
  Loaded: loaded (/usr/lib/systemd/system/elasticsearch.service; enabled; vendor preset:
  Drop-In: /etc/systemd/system/elasticsearch.service.d
           override.conf
  Active: active (running) since Sat 2022-01-08 20:35:28 HKT; 4 days ago
  Docs:
  Main PID: 11660 (java)
  CGroup: /system.slice/elasticsearch.service
           11660 /usr/share/elasticsearch/jdk/bin/java -Des.networkaddress.cache.ttl=6
           -Des.networkaddress.cache.negative.ttl=10 -XX...
           11755 /usr/share/elasticsearch/modules/x-pack-ml/platform/linux-x86_64/bin/
  Jan 08 20:35:12 es75-01 systemd[1]: Starting Elasticsearch...
  Jan 08 20:35:13 es75-01 elasticsearch[11660]: OpenJDK 64-Bit Server VM warning: Option
  Jan 08 20:35:28 es75-01 systemd[1]: Started Elasticsearch.
  Hint: Some lines were ellipsized, use -l to show in full.
[root@es75-01 ~]#
```

9. Stop Elasticsearch service using `systemctl stop elasticsearch` command.

Configure 3 nodes Elasticsearch cluster for Swarm

Create data folder for Swarm Elasticsearch

1. Create a data folder for Swarm Elasticsearch on all nodes `mkdir /var/spool/esdata`. This folder **must not** be on the boot partition to verify Elasticsearch cannot fill the boot partition.
2. Change ownership of recently created folder using `chown elasticsearch:elasticsearch /var/spool/esdata`.
3. Configure the firewall to allow Elasticsearch `vi /usr/lib/firewalld/services/elasticsearch.xml`.

```
<?xml version="1.0" encoding="utf-8"?>
<service>
  <short>Elasticsearch</short>
  <description>Elasticsearch server REST API, which is based on http traffic.</description>
  <port protocol="tcp" port="9100"/>
  <port protocol="tcp" port="9200"/>
  <port protocol="tcp" port="9300"/>
</service>
```

4. Restart the firewall service using `systemctl restart firewalld` command.
5. Add Elasticsearch firewall rule (permanently) using `firewall-cmd --permanent --add-service elasticsearch`.
6. Reload the firewall service `firewall-cmd --reload`.

Configure Elasticsearch Cluster

The following information is about an example deployment site used to configure 3 nodes Elasticsearch cluster:

Role	Application Network	Private Network	vCPU	RAM (GB)	Disk Size (GB)	Remark
ES75-01	172.16.33.21	192.168.9.21	4	8	100	Elasticsearch 7.5.2 Cluster (SSD)
ES75-02	172.16.33.22	192.168.9.22	4	8	100	Elasticsearch 7.5.2 Cluster (SSD)
ES75-03	172.16.33.23	192.168.9.23	4	8	100	Elasticsearch 7.5.2 Cluster (SSD)

1. Run the Swarm Elasticsearch configuration script `/usr/share/caringo-elasticsearch-search/bin /configure_elasticsearch_with_swarm_search.py` on the first Elasticsearch node (e.g., es75-01).
2. Enter the Elasticsearch cluster name (e.g., swarm-es75).

```
Checking ES version...
elasticsearch 7 detected
=====
About to prompt for Elasticsearch config information, config=None
Prompting admin for configuration input
Enter Elasticsearch cluster name [A string]: swarm-es75
```

3. Enter the list of all Elasticsearch server names or private network IP addresses.

```
Enter List of all the Elasticsearch server names in cluster [Comma-separated list of DNS-re:
```

4. Enter the name of the current Elasticsearch node.

```
Enter this Elasticsearch node's name [A string name from the list entered above]: 192.168.9
```

The configuration script generates a custom Elasticsearch configure file for each node in the Elasticsearch cluster:

- a. The other nodes' files are `/etc/elasticsearch/elasticsearch.yml.<node-ip>`
- b. The current node's file is `/etc/elasticsearch/elasticsearch.yml`

```
[root@elasticsearch75-04 Elasticsearch]# ll /etc/elasticsearch/
total 56
-rw-rw----. 1 root elasticsearch 199 Jan 13 11:50 elasticsearch.keystore
-rwxr-xr--. 1 root elasticsearch 3549 Jan 13 11:57 elasticsearch.yml
-rwxr-xr--. 1 root root 3355 Jan 13 11:57 elasticsearch.yml.192.168.9.22
-rwxr-xr--. 1 root root 3355 Jan 13 11:57 elasticsearch.yml.192.168.9.23
-rw-rw----. 1 root elasticsearch 2847 Jan 15 2020 elasticsearch.yml-2022-01-13T11:57
-rw-rw-r--. 1 root elasticsearch 2436 Jan 13 11:57 jvm.options
-rw-rw----. 1 root elasticsearch 2276 Jan 15 2020 jvm.options-2022-01-13T11:57:17.ba
-rw-rw----. 1 root elasticsearch 17545 Jan 15 2020 log4j2.properties
-rw-rw----. 1 root elasticsearch 473 Jan 15 2020 role_mapping.yml
-rw-rw----. 1 root elasticsearch 197 Jan 15 2020 roles.yml
-rw-rw----. 1 root elasticsearch 0 Jan 15 2020 users
-rw-rw----. 1 root elasticsearch 0 Jan 15 2020 users_roles
[root@elasticsearch75-04 Elasticsearch]#
```

5. Update Elasticsearch configure file of all Elasticsearch nodes to change the data directory path.`data: /var/lib`
`/elasticsearch -> path.data: /var/spool/esdata`. This guarantees that stored Elasticsearch data does not fill up the boot partition.

Apply changes to other nodes' configure files.

```
vi /etc/elasticsearch/elasticsearch.yml
# ===== Elasticsearch Configuration =====
# Caringo (elasticsearch 7) render date: 2022-01-13T03:51:17Z
#
# NOTE: Elasticsearch comes with reasonable defaults for most settings.
#       Before you set out to tweak and tune the configuration, make sure you
#       understand what are you trying to accomplish and the consequences.
#
# The primary way of configuring a node is via this file. This template lists
# the most important settings you may want to configure for a production cluster.
#
# Please consult the documentation for further information on configuration options:
# https://www.elastic.co/guide/en/elasticsearch/reference/index.html
#
# ----- Cluster -----
#
# Use a descriptive name for your cluster:
#
cluster.name: swarm-es75
#
# ----- Node -----
#
# Use a descriptive name for the node:
#
```

```

node.name: 192.168.9.21
#
# Add custom attributes to the node:
#
#node.attr.rack: r1
#
# ----- Paths -----
#
# Path to directory where to store the data (separate multiple locations by comma):
#
path.data: /var/spool/esdata
#
# Path to log files:
#
path.logs: /var/log/elasticsearch
#
# ----- Memory -----
#
# Lock the memory on startup:
#
bootstrap.memory_lock: true
#
# Make sure that the heap size is set to about half the memory available
# on the system and that the owner of the process is allowed to use this
# limit.
#
# Elasticsearch performs poorly when the system is swapping the memory.
#
# Increasing from default 200 but it might not really help if there are not enough nodes to
thread_pool.write.queue_size: 1000
#
# ----- Network -----
#
# Set the bind address to a specific IP (IPv4 or IPv6):
#
#network.host: 0.0.0.0
network.host: 192.168.9.21
#
# Set a custom port for HTTP:
#
#http.port: 9200
#
# For more information, consult the network module documentation.
#
# ----- Discovery -----
#
# Pass an initial list of hosts to perform discovery when this node is started:
# The default list of hosts is ["127.0.0.1", "::1"]
#
discovery.seed_hosts: ["192.168.9.21", "192.168.9.22", "192.168.9.23"]
#
# Bootstrap the cluster using an initial set of master-eligible nodes:
#
cluster.initial_master_nodes: ["192.168.9.21", "192.168.9.22", "192.168.9.23"]
#
# For more information, consult the discovery and cluster formation module documentation.
#
# By default nodes can be both master or data but if set will be preserved
#
node.master: true
node.data: true
#
# This should be set to 1 in Production
#
node.max_local_storage_nodes: 1
    
```

```
#
# ----- Gateway -----
#
# Block initial recovery after a full cluster restart until N nodes are started:
#
gateway.recover_after_nodes: 2
#
gateway.expected_nodes: 3
#
# For more information, consult the gateway module documentation.
#
# ----- Various -----
#
# Require explicit names when deleting indices:
#
#action.destructive_requires_name: true
#
# Disable automatic index creation, except for csmeter indices, Swarm NFS connectors, and ES
action.auto_create_index: "+csmeter*,+*_nfsconnector,.watches,.triggered_watches,.watcher-h:"
```

6. Copy the generated Elasticsearch configure file to other nodes.

```
scp /etc/elasticsearch/elasticsearch.yml.192.168.9.22 root@192.168.9.22:/etc/elasticsearch/
scp /etc/elasticsearch/elasticsearch.yml.192.168.9.23 root@192.168.9.23:/etc/elasticsearch/
```

7. SSH to other nodes of the Elasticsearch cluster to run the configuration script.

a. On ES75-02 node,

```
ssh root@192.168.9.22
/usr/share/caringo-elasticsearch-search/bin/configure_elasticsearch_with_swarm_search.
/etc/elasticsearch/elasticsearch.yml.192.168.9.22
exit
```

b. On ES75-03 node,

```
ssh root@192.168.9.23
/usr/share/caringo-elasticsearch-search/bin/configure_elasticsearch_with_swarm_search.
/etc/elasticsearch/elasticsearch.yml.192.168.9.23
exit
```

8. Start Elasticsearch services one by one using `systemctl start elasticsearch` command. Verify the Elasticsearch service has started completely before starting another node.

9. Verify the status of the Elasticsearch cluster. Verify the statuses for all 3 nodes of Elasticsearch Cluster are 'green'.

```
curl -iL 'http://192.168.9.2{1,2,3}:9200/_cluster/health?pretty'

[1/3]: http://192.168.9.21:9200/_cluster/health?pretty --> <stdout>
--curl--http://192.168.9.21:9200/_cluster/health?pretty
HTTP/1.1 200 OK
content-type: application/json; charset=UTF-8
content-length: 465

{
  "cluster_name" : "swarm-es75",
  "status" : "green",
  "timed_out" : false,
  "number_of_nodes" : 3,
  "number_of_data_nodes" : 3,
  "active_primary_shards" : 12,
  "active_shards" : 24,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 0,
  "delayed_unassigned_shards" : 0,
  "number_of_pending_tasks" : 0,
  "number_of_in_flight_fetch" : 0,
```

```

    "task_max_waiting_in_queue_millis" : 0,
    "active_shards_percent_as_number" : 100.0
  }

[2/3]: http://192.168.9.22:9200/_cluster/health?pretty --> <stdout>
--_curl_--http://192.168.9.22:9200/_cluster/health?pretty
HTTP/1.1 200 OK
content-type: application/json; charset=UTF-8
content-length: 465

{
  "cluster_name" : "swarm-es75",
  "status" : "green",
  "timed_out" : false,
  "number_of_nodes" : 3,
  "number_of_data_nodes" : 3,
  "active_primary_shards" : 12,
  "active_shards" : 24,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 0,
  "delayed_unassigned_shards" : 0,
  "number_of_pending_tasks" : 0,
  "number_of_in_flight_fetch" : 0,
  "task_max_waiting_in_queue_millis" : 0,
  "active_shards_percent_as_number" : 100.0
}

[3/3]: http://192.168.9.23:9200/_cluster/health?pretty --> <stdout>
--_curl_--http://192.168.9.23:9200/_cluster/health?pretty
HTTP/1.1 200 OK
content-type: application/json; charset=UTF-8
content-length: 465

{
  "cluster_name" : "swarm-es75",
  "status" : "green",
  "timed_out" : false,
  "number_of_nodes" : 3,
  "number_of_data_nodes" : 3,
  "active_primary_shards" : 12,
  "active_shards" : 24,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 0,
  "delayed_unassigned_shards" : 0,
  "number_of_pending_tasks" : 0,
  "number_of_in_flight_fetch" : 0,
  "task_max_waiting_in_queue_millis" : 0,
  "active_shards_percent_as_number" : 100.0
}

```

Refer to [Configuring Elasticsearch](#) to update environment and log settings. Perform a rolling restart of Elasticsearch services to apply changes once all updates are made.

Only for new setup of Elasticsearch cluster

Refer to the below steps if Elasticsearch nodes are unable join the cluster and return 'master not discovered exception' error:

1. Stop Elasticsearch service on all nodes.
2. Delete files/folders inside `/var/spool/esdata/` on all nodes.
3. Restart Elasticsearch nodes one by one.

✔ Next, [Swarm gateway](#).

Swarm Gateway

- [Overview](#)
- [Prerequisites](#)
 - [Setup CentOS for Cloud Gateway](#)
- [Install Swarm Gateway](#)
- [Cluster Initialization](#)

Overview

For more information about Swarm Gateway, its concepts, and operations, see [Swarm Content Gateway](#).

Prerequisites

- CentOS 7.x Setup
- [Swarm 14.0 bundle package](#)

Setup CentOS for Cloud Gateway

Below is an example deployment for the site.

Minimal Server	Host name	
	cg77-01.datacore.internal	cg77-02.datacore.internal
Application Network	IP Address	
	172.16.33.16/24	172.16.33.17/24
Gateway	172.16.33.1	
NTP	172.16.33.10	

To set up time synchronization for CentOS 7, see [here](#).

Install Swarm Gateway

i Info

Creating a user group/user/password applies when PAM authentication is used. Create a user group via standard Linux user administration commands.

1. Create a user group

```
groupadd clusteradmins
```

2. Add a new user.

```
adduser dcadmin -g clusteradmins -p datacore
passwd dcadmin
```

3. Update CentOS and install EPEL.

```
yum -y update
yum -y install epel-release
```

4. Install NTP server and verify NTP starts on system boot.

```
yum -y install chrony
systemctl enable chronyd
systemctl start chronyd
```

5. Install Java 8 JDK.

```
yum -y install java-1.8.0-openjdk
```

6. Configure the firewall to allow Swarm Gateway.

```
vi /usr/lib/firewalld/services/swarm_gateway.xml
```

```
<?xml version="1.0" encoding="utf-8"?>
<service>
  <short>SwarmGateway</short>
  <description>Gateway server, which is based on http traffic.</description>
  <port protocol="tcp" port="80"/>
  <port protocol="tcp" port="91"/>
  <port protocol="tcp" port="8090"/>
  <port protocol="tcp" port="9100"/>
</service>
```

7. Restart the firewall service.

8. Add Swarm Gateway firewall rule (permanent)

```
firewall-cmd --permanent --add-service swarm_gateway command
```

9. Reload the firewall service .

```
firewall-cmd --reload
```

10. Unzip the Swarm 14.0 bundle package.

```
cd /root/datacore
unzip Swarm-v14.0-Update2-20220209.zip
```

11. Get the latest gateway and UI RPMs from the downloaded Swarm bundle.

```
ll ~/datacore/Gateway/
total 57352
-rw-r--r--. 1 root root 58718208 Dec 16 22:37 caringo-gateway-7.8.0-1.noarch.rpm
-rw-r--r--. 1 root root      395 Dec 16 22:37 README.txt
-rw-r--r--. 1 root root    1927 Oct  2 04:17 RPM-GPG-KEY

ll ~/datacore/UI/
total 21616
-rw-r--r--. 1 root root 2835590 Oct  2 04:17 caringo-gateway-webui-7.5.0-1.noarch.rpm
-rw-r--r--. 1 root root 19286001 Dec 16 22:37 caringo-storage-webui-3.4.0-1.noarch.rpm
-rw-r--r--. 1 root root    359 Dec 16 22:37 README.txt
-rw-r--r--. 1 root root    1927 Oct  2 04:17 RPM-GPG-KEY
```

12. Install Swarm RPM public key included with the distribution bundle.

```
rpm --import RPM-GPG-KEY
```

13. Install Cloud Gateway components.

```
yum -y install caringo-gateway-7.8.0-1.noarch.rpm
```

14. Complete the IDSYS document for PAM user authentication.

```
vi /etc/caringo/cloudgateway/idsys.json
```

```
{
  "pam": {
    "name": "idsys-pam",
    "description": "PAM identity management configuration",
    "cookieName": "token",
    "tokenPath": "/.TOKEN/",
    "tokenAdmin": "dcadmin@"
  }
}
```

15. Complete policy document for the access control.

```
vi /etc/caringo/cloudgateway/policy.json

{
  "Id": "Root policy -- grant admins full access to everything",

  "Statement": [
    {
      "Sid": "Grant admins full access",
      "Resource": "*",
      "Principal": {
        "user": ["dcadmin@"],
        "group": ["clusteradmins@"]
      },
      "Action": [ "*" ],
      "Effect": "Allow"
    }
  ]
}
```

16. Install Gateway WebUI.

```
yum -y install caringo-gateway-webui-7.5.0-1.noarch.rpm
```

17. Edit the Cloud Gateway configure file.

```
vi /etc/caringo/cloudgateway/gateway.cfg
```

- a. Set *adminDomain* for the administrative domain name to be created. It is recommended to always set a unique name for each Swarm cluster to prevent replication collision between Source and Target Swarm Cluster when using replication feeds. Set a unique name for each Swarm cluster to prevent replication collision.
- b. Set *hosts* for the storage cluster nodes (IP address of storage nodes).
- c. Set *indexerHosts* to Elasticsearch Servers (IP address of Elasticsearch nodes).
- d. Enable S3.
- e. Enable *Cluster Admin* on the gateway server to allow usage of the Swarm Storage UI.
- f. Set Management Port to 91. The admin user and password must be the same as configured in SCS.
 - i. Set *managementUser* to the same value as *admin.username*
 - ii. Set *managementPassword* to the same value as *admin.password*

```
# Following are only needed when enabling cluster_admin
managementPort = 91
managementUser = dcadmin
managementPassword = datacore
```

- g. Enable **metering**.

Below is a sample Gateway configuration file in its entirety.

```
#
# gateway.cfg -- configuration file for Caringo CloudScaler Gateway server
#
# Please read the CloudScaler Gateway Administration Guide for an explanation
# of the parameters in this configuration file.
#
# Basic Configuration Steps:
# -----
```

```

#
# 1. Set '[gateway]adminDomain' to administrative domain's name
#
# 2. Use the Swarm static locator for the backend cluster:
#     - Set '[storage_cluster]locatorType' to 'static'
#     - Set '[storage_cluster]hosts' for your Swarm nodes
#
# 3. Configure the Elasticsearch metadata backend:
#     - Set '[storage_cluster]indexerHosts' to the metadata search servers
#
# 4. If the S3 protocol is used:
#     - Ensure that bindAddress:bindPort does not conflict with SCSP
#
# NOTE: For production use, you will need to adjust the thread counts,
#       connection limits, connection time outs, and the space/limits
#       for the HTTP multi-part spool.
#
#
# Client communications and handling
#
[gateway]

adminDomain = Lab1GatewayAdminDomain
threads = 200
# multipartSpoolDir = /var/spool/cloudgateway
# multipartUsageAllowed = 50
# sanitizeErrors = false
# enablePasswordEncryption = false
# legacyOnlyMode = false

#
# Storage cluster back-end configuration
#
[storage_cluster]

locatorType = static
hosts = 192.168.9.143 192.168.9.144 192.168.9.145
# port = 80
# dataProtection = immediate
# blockUndeletableWrites = true

indexerHosts = 192.168.9.21 192.168.9.22 192.168.9.23
# indexerPort = 9200
# indexerSocketTimeout = 120

# maxConnectionsPerRoute = 100
# maxConnections = 250
# connectTimeout = 60
# socketTimeout = 120
# idleTimeout = 120
# continueWaitTimeout = 30

# Following are only needed when enabling cluster_admin
managementPort = 91
managementUser = dcadmin
managementPassword = datacore

#
# SCSP front-end protocol
#
[scsp]

enabled = true
    
```

```
bindAddress = 0.0.0.0
bindPort = 80

# Set the following if this Gateway is the target of a push-style replication feed.
# See 'Replication Feeds' in the Swarm documentation for details.
# allowSwarmAdminIP = list,of,node,ips -or- all
allowSwarmAdminIP = all

# Set these to the external ports on your proxy, if using one
#externalHTTPPort = 80
#externalHTTPSPort = 443

#
# S3 front-end protocol
#
[s3]

enabled = true
bindAddress = 0.0.0.0
bindPort = 8090

# Set these to the external ports on your proxy, if using one
#externalHTTPPort = 80
#externalHTTPSPort = 443

#
# Cluster management protocol
#
[cluster_admin]

enabled = true
bindAddress = 0.0.0.0
bindPort = 91
# secretKey = yoursecretkeyhere

# Set these to the external ports on your proxy, if using one
#externalHTTPPort = 91
#externalHTTPSPort = 1443

#
# Internal "folder listing service" config
#
[folder_listings]

# enhancedListingConsistency = true      # force ES index flush before query
# feedCheckInterval = 60000              # in milliseconds, how often to check for non-searchable
# suppressNonSearchableError = false     # suppress 412 error when searching non-searchable domain

#
# Caching timers : 0 == disabled
#
[ caching ]

# authRefresh = 300
# tokenRefresh = 300
# idsysRefresh = 300
# policyRefresh = 300
# xformRefresh = 300
# metadataRefresh = 300
# domainExistenceRefresh = 300

#
# Metering Support
#
[ metering ]
```

```

enabled = true
# flushIntervalSeconds = 300
# retentionDays = 100
# storageSampleIntervalSeconds = 3600

#
# Quota Support
#
[quota]

enabled = false
smtpHost = localhost
mailFrom = donotreply@localhost

# mailSubjectTemplate = Quota state change notification
# mailTemplate = Metric %metric% changed to %state% state in %contextType% %contextName%.

# smtpPort = 25
# smtpUser =
# smtpPassword =

# minRefreshDeadline = 60
# maxRefreshDeadline = 3600
# numRefreshThreads = 4
# maxRefreshRetries = 3
# maxQueueSize = 10000
# queryTTL = 3600
# refreshRetryDelay = 10
# refreshIdleSleep = 3

#
# Prometheus metrics capturing support
#
[metrics]

metricsEnabled = true      # default enabled
# metricsPort = 9100      # port where to scrape for metrics

#
# Remote synchronous write support (RSW)
#
[rsw]

# enabled = true          # default enabled
# maxWaitTime = -1       # timeout in seconds to wait for RSW completion, -1 means no timeout
# enableInfoLogging = true # default true, log RSW operations at INFO level

#
# Object Lock Support (Retention, LegalHold)
#
[object_locking]

# scspDeleteUsesS3Logic = true # Allows delete of a locked current object version (via delete ma
# retentionMaxYears = 100     # Max retention duration
    
```

Cluster Initialization

The Gateway must be initialized to guarantee a proper runtime environment once the configuration is complete. This guarantees the administrative domain exists within the storage cluster and also secures all plaintext passwords entered into the configuration files.

1. Verify Gateway is not running.

```
systemctl stop cloudgateway
```

2. Perform the initialization of the environment.

```
/opt/caringo/cloudgateway/bin/initgateway
```

3. Start Cloud gateway and verify the gateway service is running on system boot once initialization has successfully completed.

```
systemctl start cloudgateway  
systemctl enable cloudgateway
```

✔ Next, [setup additional gateways](#).

Setup Additional Gateways

Mandatory to initialize first gateway

It is assumed that the first gateway has been fully configured and initialized. Refer to [cluster initialization](#) and then setup additional gateways using the below steps if not:

1. Repeat [install gateway](#) on the additional machines by referring to steps 1 to 13.
2. Copy configure files from 1st Gateway before starting gateway services.

```
scp root@172.16.33.17:/etc/caringo/cloudgateway/gateway.cfg /etc/caringo/cloudgateway/gatew:
scp root@172.16.33.17:/etc/caringo/cloudgateway/idsys.json /etc/caringo/cloudgateway/idsys.:
scp root@172.16.33.17:/etc/caringo/cloudgateway/policy.json /etc/caringo/cloudgateway/polic;
```

3. Start gateway services and verify they are running on the system boot.

```
systemctl start cloudgateway
systemctl enable cloudgateway
```

Info

No further gateway initialization is needed for these additional Gateway instances.

Next, [install Swarm storage UI](#).

Install Swarm Storage Management UI

Install Swarm Management UI on one or multiple gateways depending on requirements. Refer to the following steps to install it on a single gateway:

1. Install Swarm Storage UI on one of the gateway servers (e.g., gw77-01 - 172.16.33.16) using `yum -y install caringo-storage-webui-3.4.0-1.noarch.rpm`. The gateway services restart automatically.
2. Navigate to the login page of Storage UI to configure the search feed. For an example storage cluster, the URL is `http://172.16.33.16:91/_admin/storage`.

✔ Next, [add search feed](#).

Add Search Feed

Add search feeds to the Swarm cluster by referring to the following steps:

1. Log in to Swarm Storage Management UI. For the example cluster, this URL is: `http://172.16.33.16:91/_admin/storage`



Use the administrator user name and password provided when configuring SCS.

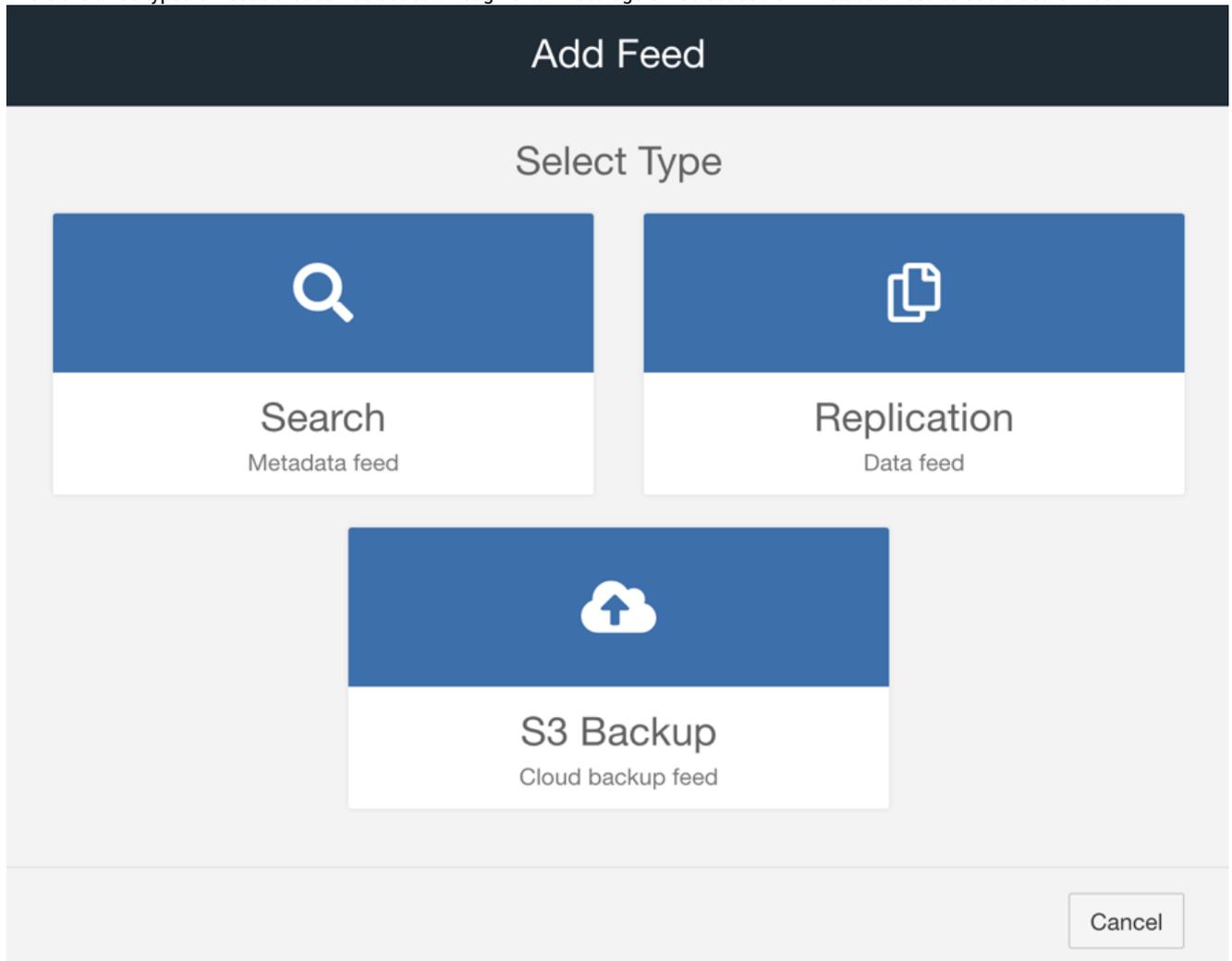
2. Select *Storage Management*.



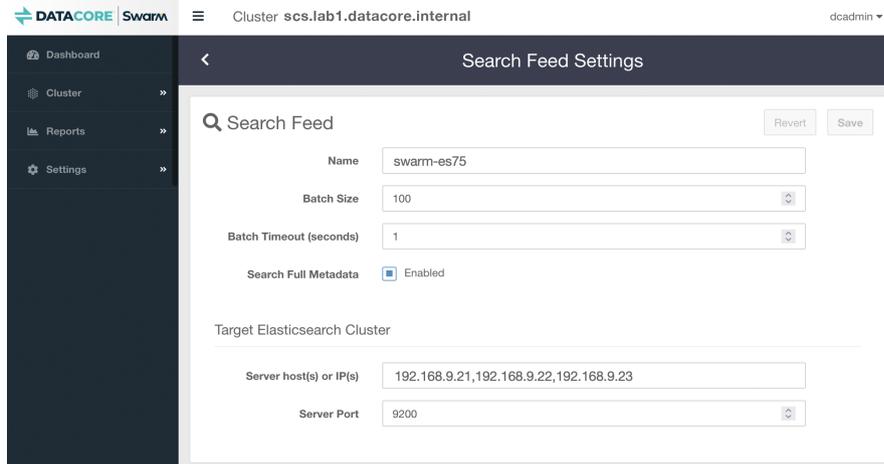
3. Navigate to *Cluster > Feeds* from the left navigation pane.



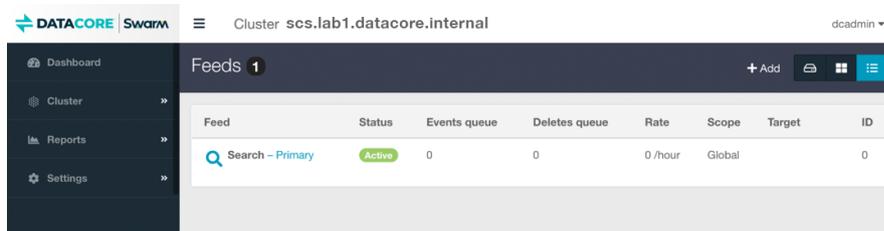
4. Click *+Add*.
5. There are three types of feeds that can be added through Swarm Storage UI. Select *Search - Metadata feed* to add a Search feed.



6. Enter a Search feed name and IP addresses of the target Elasticsearch cluster.



7. Click **Save** to create a Primary feed as shown below:



The Swarm cluster begins populating metadata to the Elasticsearch Cluster.

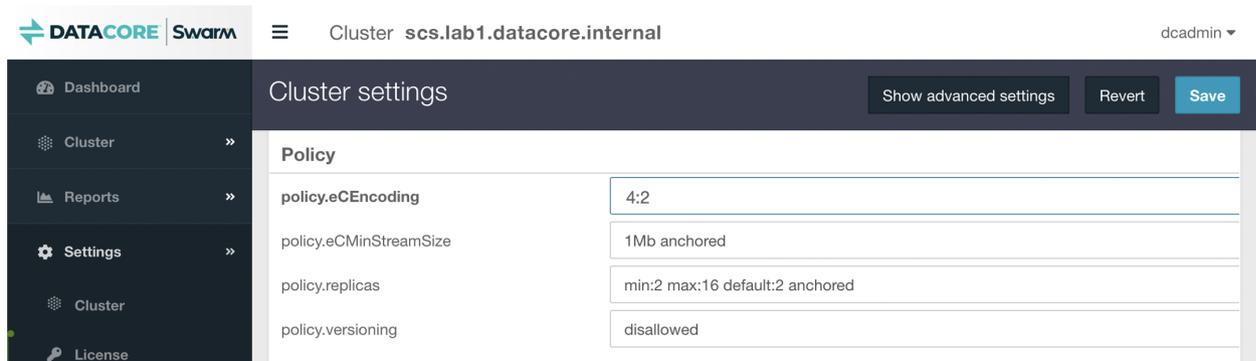
Info
The progress can be monitored on Dashboard within the UI.

Next, [Erasure Coding \(EC\) ratio configuration](#).

Erasure Coding Ratio Configuration

The default Erasure Coding configuration comes by default in a cluster. Refer to the following steps for any custom [Erasure Coding configuration](#) needed in a cluster:

1. Log in to Swarm Storage UI.
2. Navigate to *Settings > Cluster* from the left navigation pane.
3. Scroll down to the Policy section and update the value for the `policy.ecEncoding`.



Total number of required nodes are calculated using $(k + p) / p$ formula where k denotes the number of data segments and p denotes parity. Based on this formula, administrators can determine the number of nodes required using the default EC protection level of "node".

For example, a 4:2 (k:p) encoding yields:

$$(4 + 2) / 2 = 3 \text{ nodes required}$$

Data is still fully protected in the event 1 node out of a 4 node cluster is offline. See [Implementing EC Encoding Policy](#) for details.

✔ Next, [creating tenant, domain and bucket](#).

Create new Tenant, Domain and Bucket

Refer to [Configuring Tenants](#) to create a new tenant.

Refer to [Configuring Domains](#) to create a new domain.

Refer to [Configuring Buckets](#) to create a new bucket.

✔ Next, [domain level replication](#).

Domain-level Replication

See [Replication Feeds](#) for details about replication and how to add a replication feed.

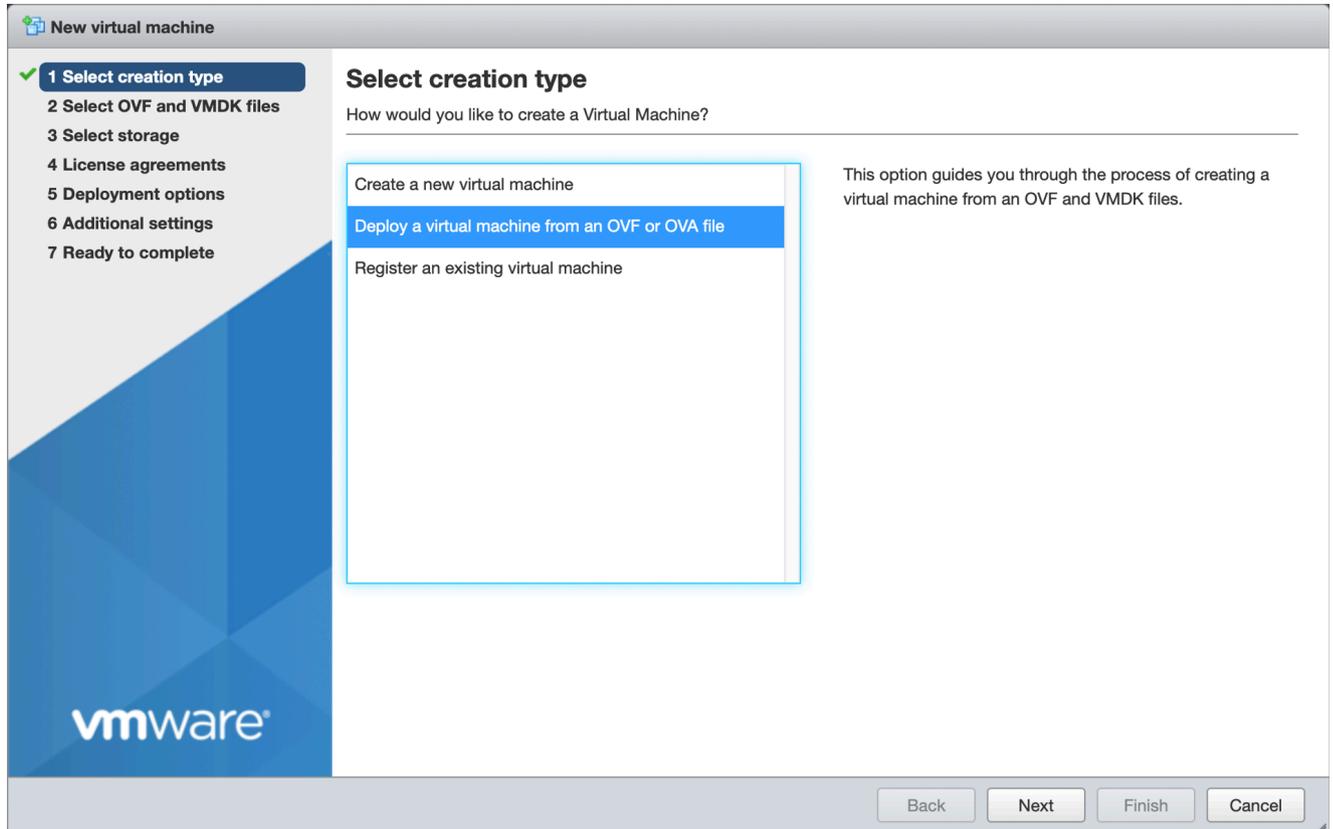
✔ Next, [Swarm Telemetry](#).

Swarm Telemetry

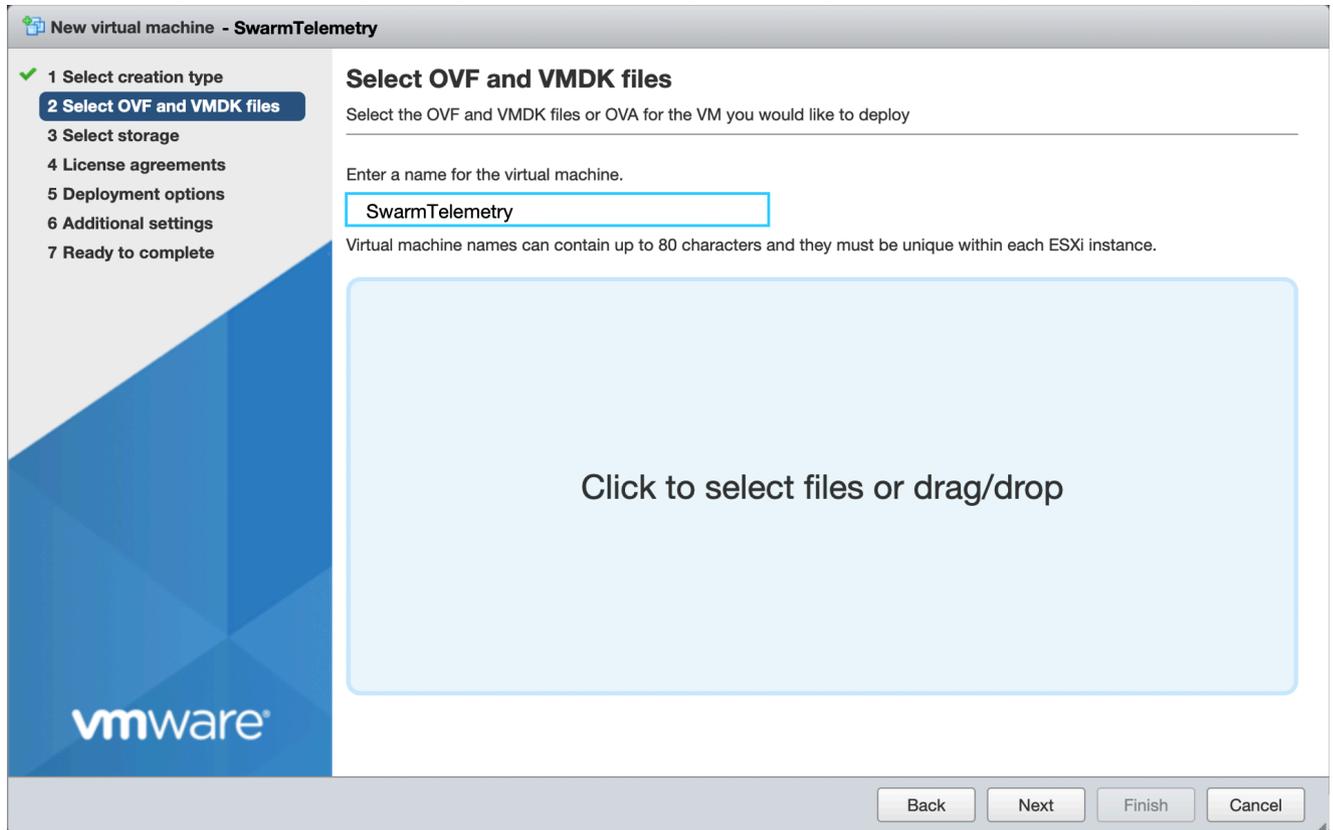
Clients must contact DataCore support to download Swarm Telemetry (OVA for VMware ESXi). To set up time synchronization for CentOS 7, see [here](#).

Refer to the following steps to deploy Swarm Telemetry (the values shown are for illustration purposes) once downloaded:

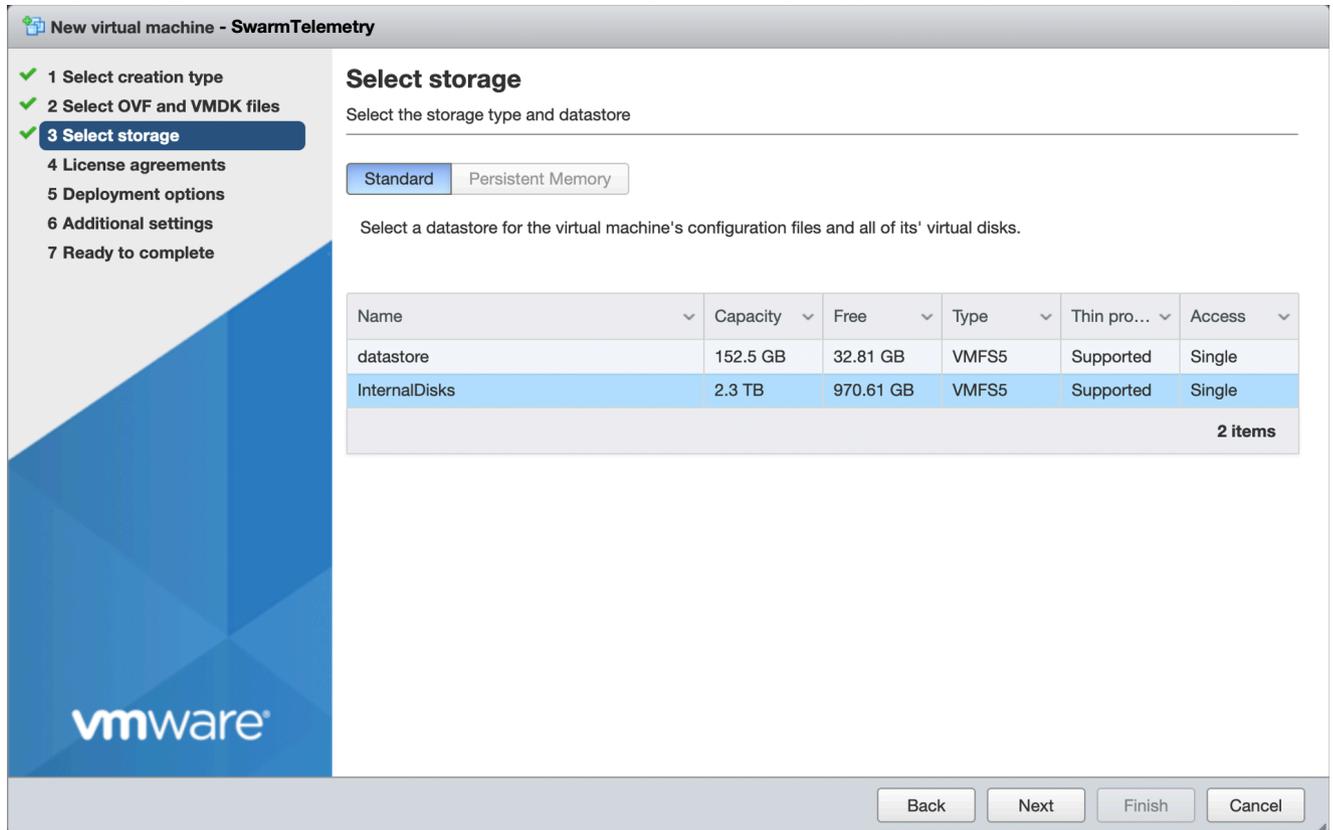
1. Log in to VMware ESXi and click *Create/Register VM*.
2. Select *Deploy a virtual machine from an OVF or OVA file* and click *Next*.



3. Drag and drop *swarmTelemetry-v12.1.0-x86_64.ova* and click *Next*.

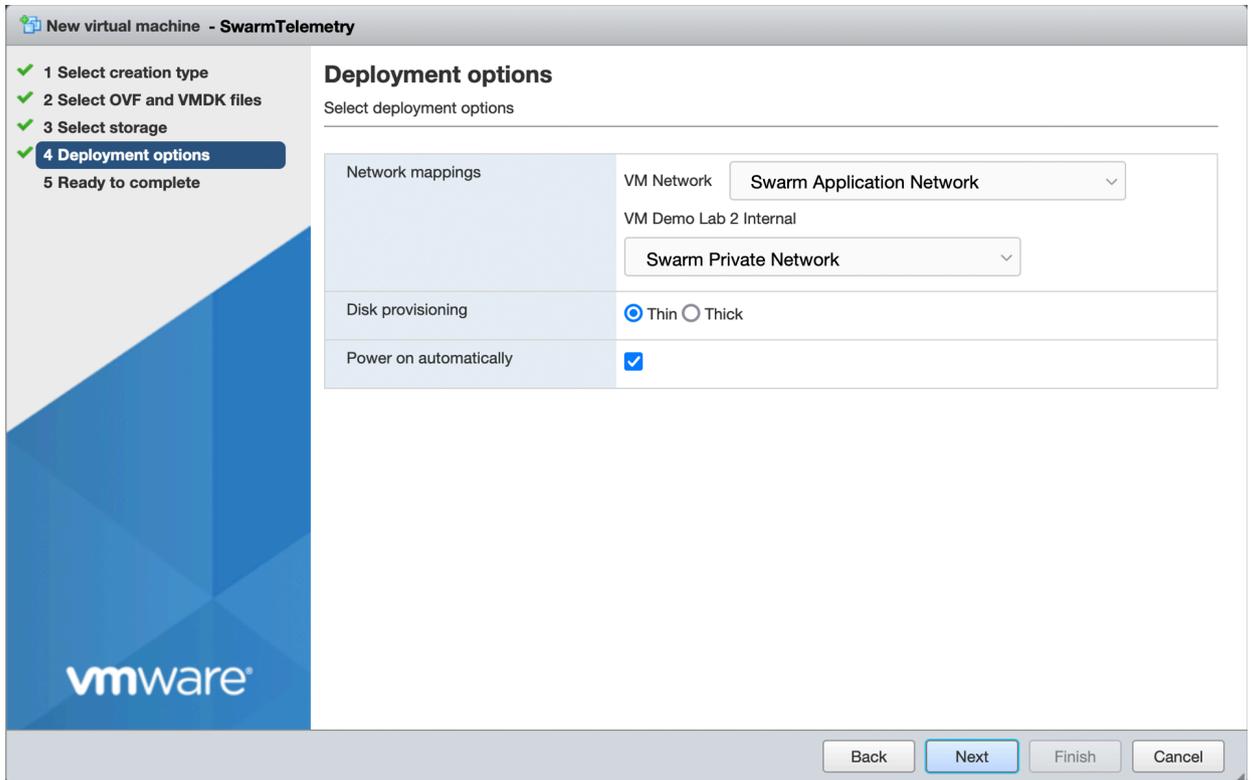


4. Select the appropriate datastore and click *Next*.

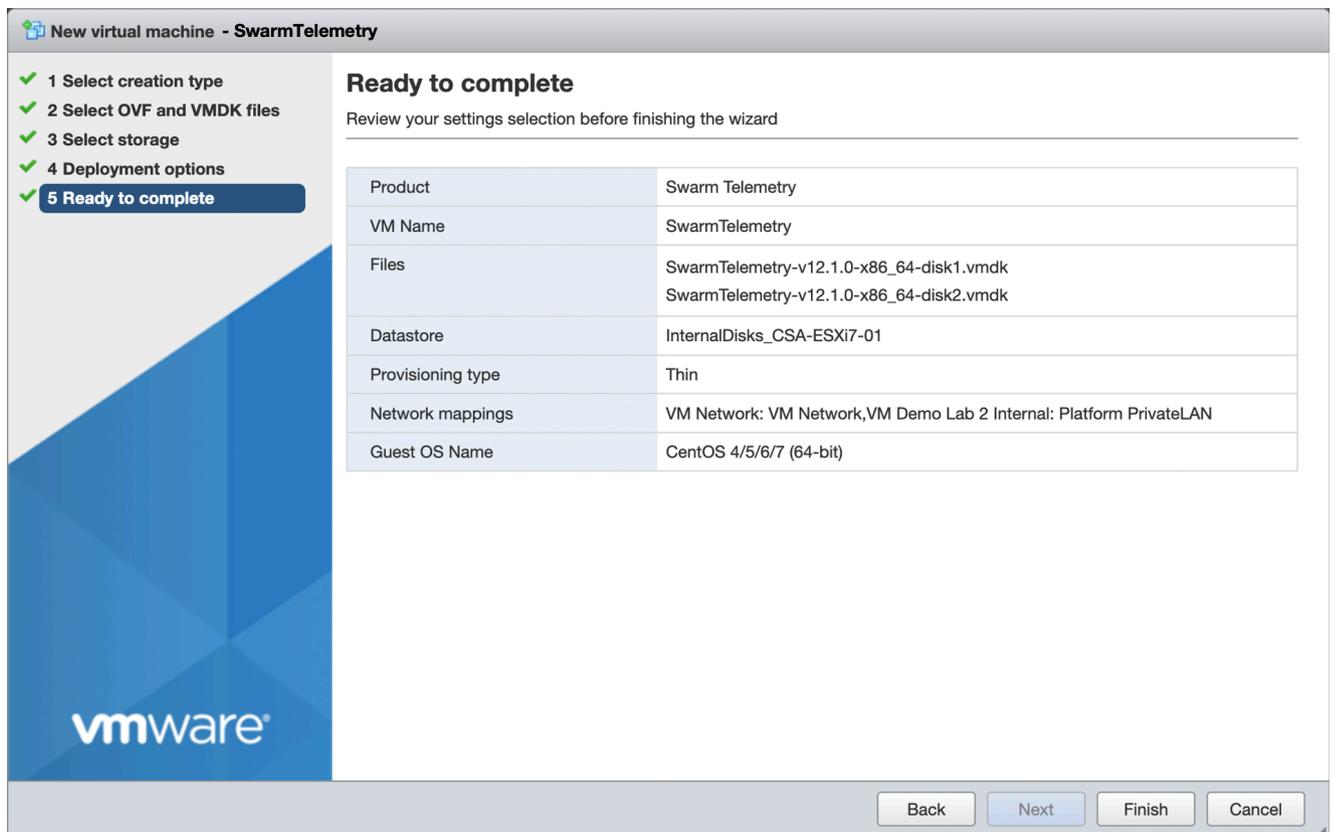


5. Select an appropriate network from ESXi, then click *Next* button.

- a. VM Network – Application network
- b. VM Demo Lab 2 Internal – Swarm Storage network



6. Verify settings before finalizing. Deploy SwarmTelemetry to ESXi if correct.



7. Power on SwarmTelemetry VM.

8. Update IP addresses from ESXi Console to enable SwarmTelemetry which accesses Gateway and Swarm storage nodes to load metrics and reboots SwarmTelemetry.

- ens33 – 172.16.33.15/24
- ens160 – 192.168.9.15/24

9. SSH/Putty to SwarmTelemetry using;

- a. Username - root
- b. Password - datacore

10. Modify Prometheus configure file by;

a. Adding gateways job

Note: It is recommended to use DNS names so IP addresses are not exposed on the dashboards.

b. Adding Swarm Nodes job

```
vi /etc/prometheus/prometheus.yml
```

```
# Add Gateways
- job_name: 'gw77-01'
  static_configs:
  - targets: ['172.16.33.16:9100']
  relabel_configs:
  - source_labels: [__address__]
    regex: "([^:]+\):\d+"
    target_label: instance

- job_name: 'gw77-02'
  static_configs:
  - targets: ['172.16.33.17:9100']
  relabel_configs:
  - source_labels: [__address__]
    regex: "([^:]+\):\d+"
    target_label: instance

- job_name: 'swarm'
  scrape_interval: 30s
  static_configs:
  - targets: ['192.168.9.143:9100', '192.168.9.144:9100', '192.168.9.145:9100']
  relabel_configs:
  - source_labels: [__address__]
    regex: "([^:]+\):\d+"
    target_label: instance
```

11. Restart Prometheus server using `systemctl restart prometheus` command.

12. Browse <http://172.16.33.15:9090/targets> URL.

Prometheus Alerts Graph Status ▾ Help Classic UI

Targets

All Unhealthy

alertmanager (1/1 up) [show less](#)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9093/metrics	UP	instance="localhost:9093" job="alertmanager"	1.256s	4.104ms	

gw77-01 (1/1 up) [show less](#)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://172.60.88.216:9100/metrics	UP	instance="172.60.88.216:9100" job="gw77-01"	19.744s	4.913ms	

gw77-02 (1/1 up) [show less](#)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://172.60.88.217:9100/metrics	UP	instance="172.60.88.216:9100" job="gw77-01"	19.744s	4.913ms	

prometheus (1/1 up) [show less](#)

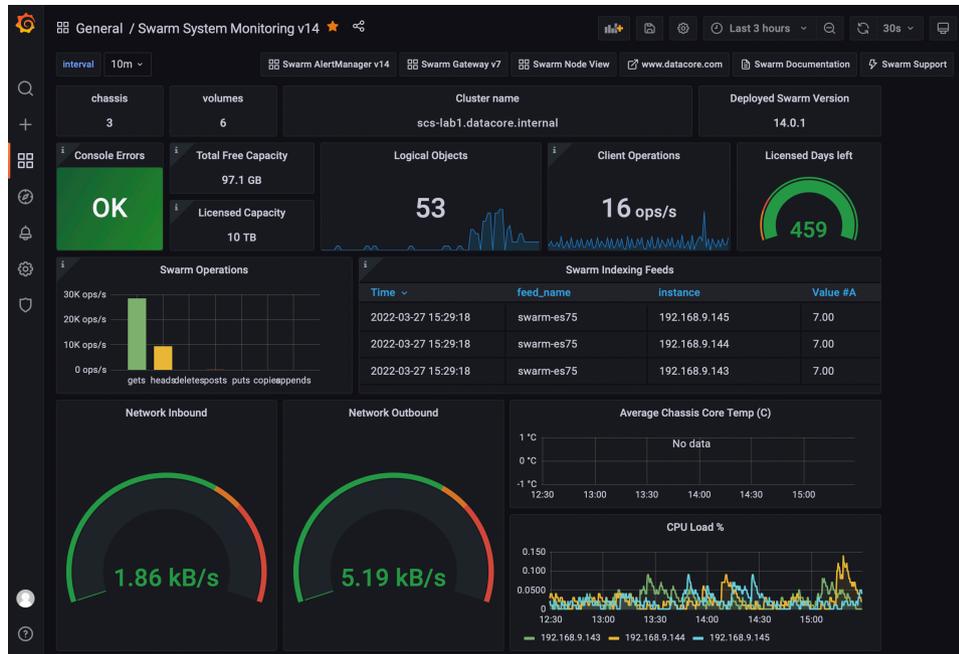
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	11.391s	5.764ms	

swarm (3/3 up) [show less](#)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://192.168.9.143:9100/metrics	UP	instance="192.168.9.143" job="swarm"	16.27s	19.392ms	
http://192.168.9.144:9100/metrics	UP	instance="192.168.9.144" job="swarm"	1.584s	18.038ms	
http://192.168.9.145:9100/metrics	UP	instance="192.168.9.145" job="swarm"	20.484s	22.449ms	

13. Log in to SwarmTelemetry Grafana using default credentials;

- a. Username - admin
- b. Password - datacore



SCS 2.0 Installation

- [Overview](#)
- [Prerequisites](#)
- [Installation Steps](#)

Overview

SCS 2.0 is a UI-based solution in a modernized form, which comes with a downloadable Pre-installer (package) available to download on the DataCore support portal. It is easy to install and deploy the SCS 2.0 ecosystem on the VMware vCenter environment. A user can configure the Swarm ecosystem based on the expected workload. The automatic installation and configuration make this ecosystem more efficient.

Prerequisites

vCenter credentials	vCenter server
	admin username
	admin password
vCenter Infrastructure details	Datacenter
	Datastore
	ESX host FQDN
	Management Network (Public network)
	Pxeboot network (Private network)
Swarm Pre-Installer package	Available to download on DataCore support portal

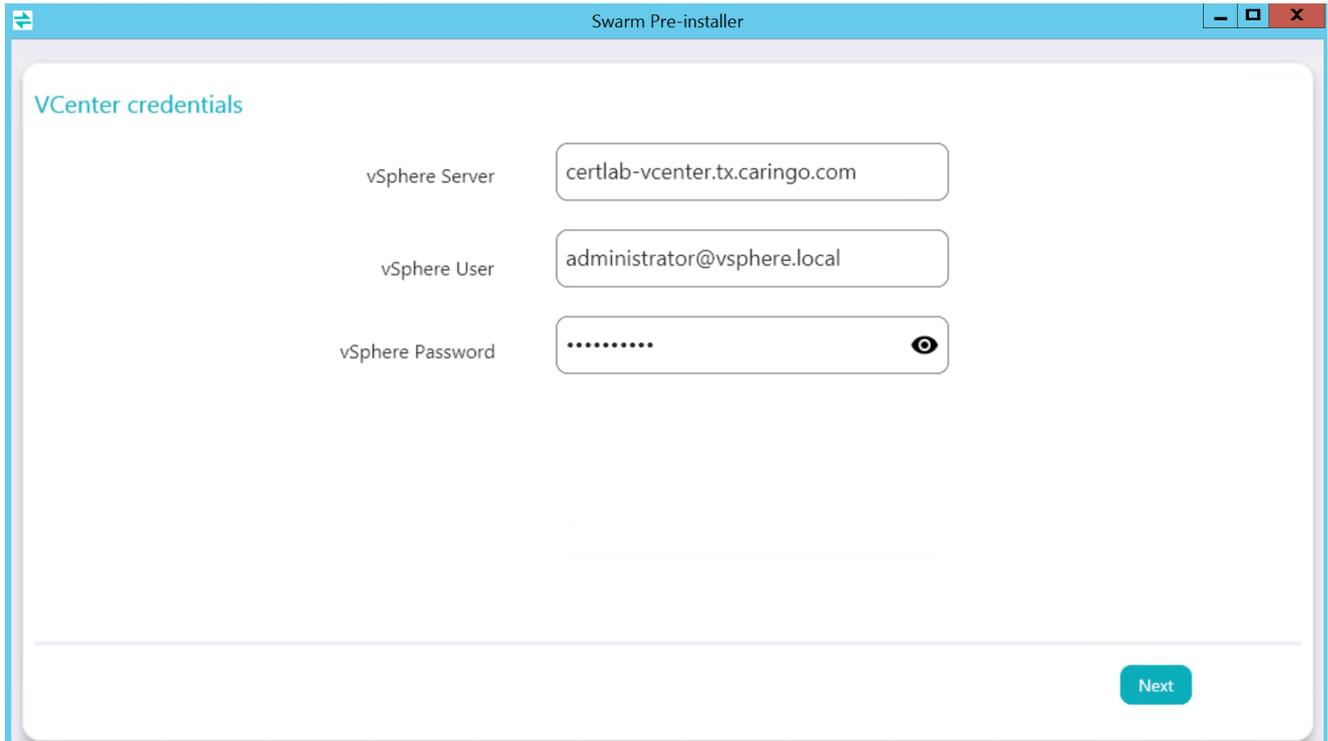
Installation Steps

Steps to install SCS 2.0 on the VMware vCenter environment are:

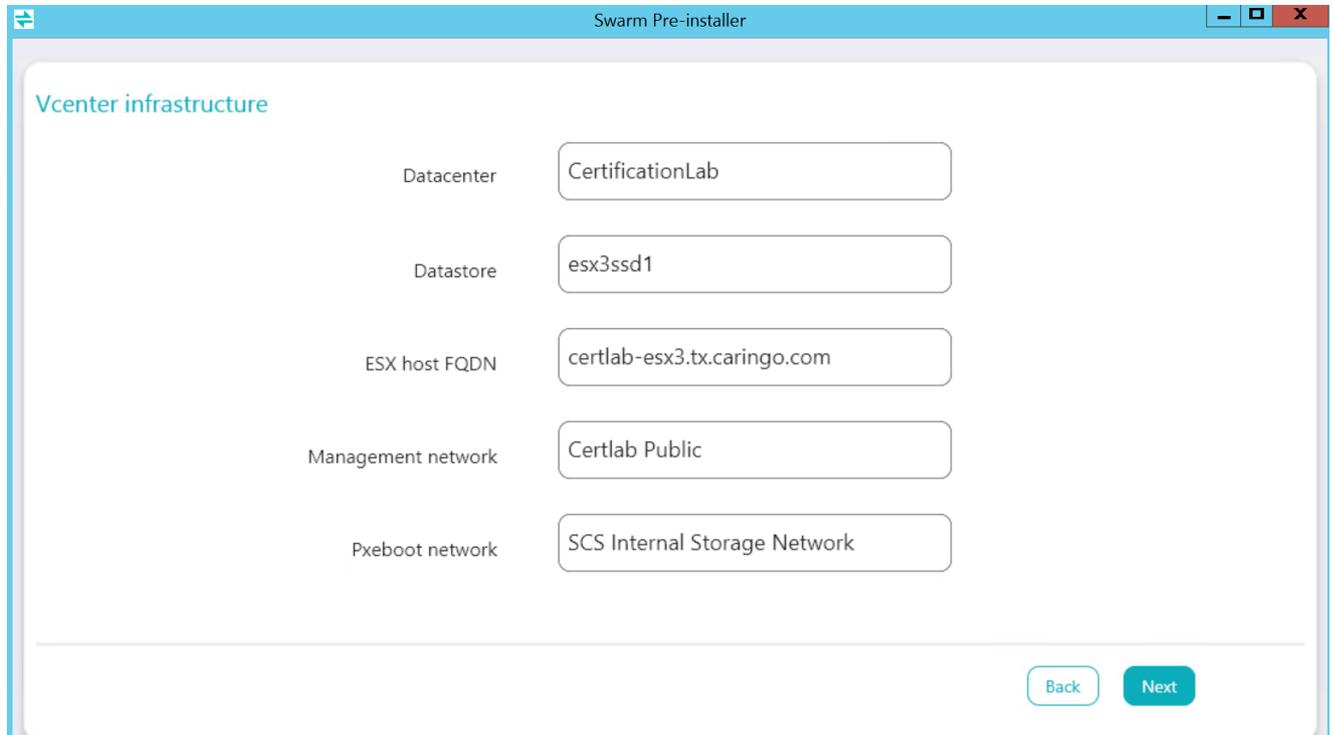
- [Install Swarm Pre-installer](#)
- [Configure SCS 2.0](#)
- [Upload YAML File and Start Storage Nodes](#)
- [Install Elasticsearch for SCS 2.0](#)
- [Install Gateway for SCS 2.0](#)
- [Install Telemetry Server](#)
- [Access Links to Dashboard](#)

Install Swarm Pre-installer

1. Download the Swarm Pre-installer from the DataCore website. It is downloaded in the zip file format.
2. Extract the package.
3. Double click on Swam Pre-installer for initiating the installation.
4. Provide vCenter credentials and click *Next*.



- a. vSphere Server - A vSphere server hostname or IP address.
 - b. vSphere User - An admin user with the access to create VMs.
 - c. vSphere Password - The admin password.
5. Provide vCenter Infrastructure details to install the SCS node then click *Next*.



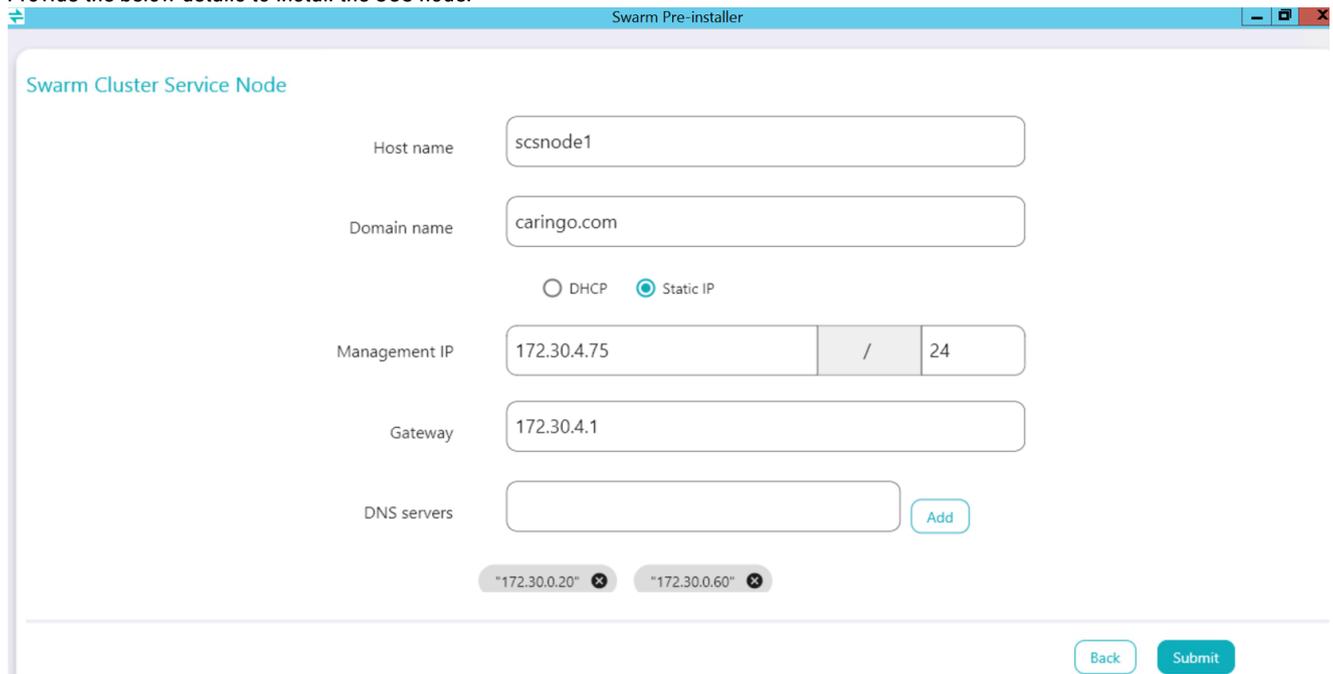
The screenshot shows the 'Vcenter infrastructure' configuration screen in the Swarm Pre-installer. It contains five input fields for configuration:

- Datcenter:** CertificationLab
- Datastore:** esx3ssd1
- ESX host FQDN:** certlab-esx3.tx.caringo.com
- Management network:** Certlab Public
- Pxeboot network:** SCS Internal Storage Network

At the bottom right, there are 'Back' and 'Next' buttons.

- a. Datcenter - Enter the data center name based on region and availability to install the SCS node.
- b. Datastore - Disk available on the host for storage.
- c. ESX host FQDN - Enter the ESX hostname.
- d. Management Network - Public network
- e. Pxeboot network - Choose a private network for Pxebooting configuration.

6. Provide the below details to install the SCS node:



The screenshot shows the 'Swarm Cluster Service Node' configuration screen in the Swarm Pre-installer. It contains the following configuration details:

- Host name:** scsnode1
- Domain name:** caringo.com
- IP Configuration:** DHCP is unselected, and Static IP is selected.
- Management IP:** 172.30.4.75 / 24
- Gateway:** 172.30.4.1
- DNS servers:** An empty input field with an 'Add' button.

At the bottom, there are two existing IP entries: "172.30.0.20" and "172.30.0.60", each with a close button. At the bottom right, there are 'Back' and 'Submit' buttons.

- a. Enter the hostname and domain name for the SCS node.

b. Select the IP as DHCP or Static IP.

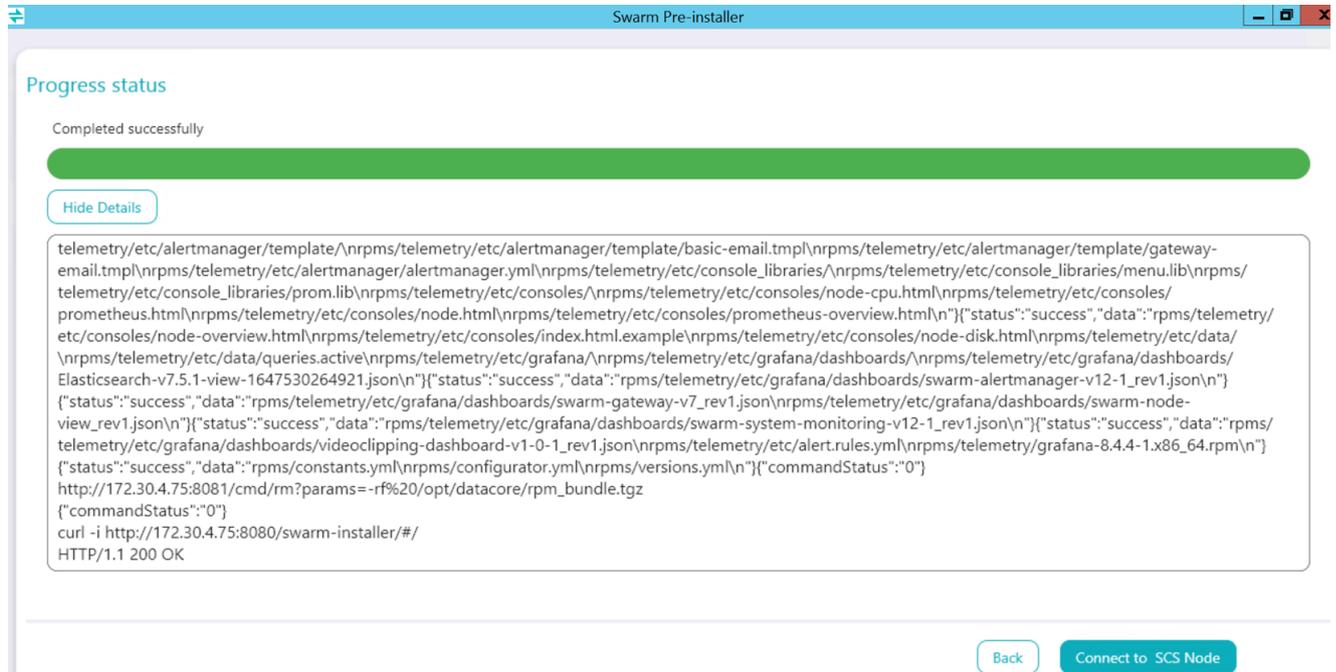
i. For static IP;

- Management IP - An IP address for the public network and its network mask.
- Gateway - IP address of the default network Gateway.
- DNS servers - Add one or multiple DNS servers depending on the requirements.

ii. For DHCP, an IP address gets assigned automatically.

7. Click *Submit*.

This creates a template VM followed by a VM for the SCS node and assigns the configured IP address. Wait for the installation to complete.

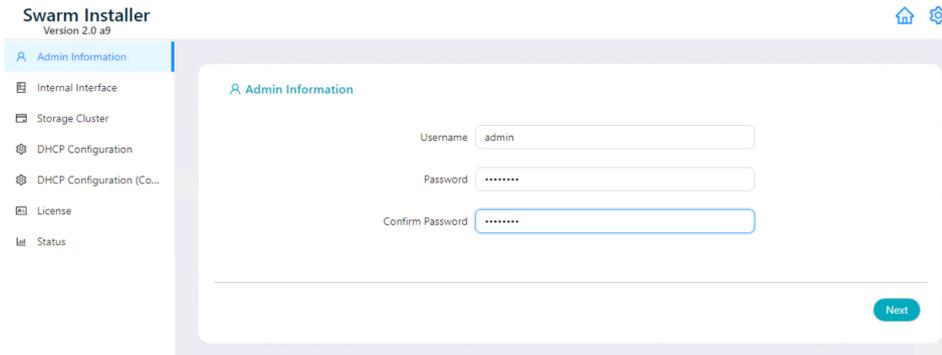


8. Click **Connect to SCS Node** to launch the Swarm Installer.

✔ Next, [Configure SCS](#).

Configure SCS 2.0

1. Provide admin information (username and password) for the SCS. Storage nodes and Gateway use this username & password when Grafana and PAM authentication is selected.



2. Click *Next*.
3. Select the network interface from the drop-down list for the internal interface. This interface is used to PXEboot the storage nodes.

Internal Interface

Internal Interface

IP Address

Network Mask

[Back](#) [Next](#)

- a. Enter the IP address and network mask for the internal interface.
4. Click *Next*.
 5. Provide a Swarm site Id; it is a unique Id given to the cluster. Site Id must be a human-oriented name used to identify a cluster among different clusters available within the organization.

Platform Server: Storage Cluster

Site Id

Storage Cluster Name

6. Swarm generates a unique cluster name using the site Id. Customers are allowed to modify it and can provide any unique name for the Swarm storage cluster.
7. Click *Next*.
8. Configure DHCP service running on the SCS node:

 **Platform Server: DHCP Configuration**

DHCP Gateway

DNS Domain Name

DNS Servers

NTP Servers

- a. DHCP Gateway IP and domain name are picked automatically based on previous steps where DHCP gateway IP is non-editable.
- b. Details of the already added DNS and NTP servers are picked, however, a customer can add more DNS/NTP servers if required.
- c. DHCP Reserve Lower & Upper - Number of IP addresses to reserve for lower and upper network subnet range out of the range defined.
Example: In case there are 235 IP addresses, out of that first 10 IP addresses are reserved for the lower range and the last 20 for the upper range.

⚙️ Platform Server: DHCP Configuration (Contd)

DHCP Reserve Lower ⓘ

DHCP Reserve Upper ⓘ

DHCP Transient % ⓘ

DHCP Reserve Default Time

DHCP Lease Maximum Time

[Back](#) [Next](#)

d. DHCP Transient % - The percentage of remaining IP addresses (which are not reserved in the lower and upper range) allocated for the transient pool.

Example: 50% of the remaining range is reserved for pooling and the rest is reserved for storage.

e. DHCP Reserve Default Time - DHCP lease expiration time in seconds if the client has not requested any time.

f. DHCP Lease Maximum Time - Maximum lease time in seconds allocated by the server when the client requests the time.

9. Click *Next*.

10. Upload the License file using *Add* button. Once the license file is uploaded or the URL to the license file is provided, click *Add*.

Add/Edit License



License URL

OR

[Upload File](#)

[Cancel](#) [Add](#)

The uploaded file is added with the below details:

 License

Add

Serial Number	Customer Name	Uploaded on	Expire date	storageMaxTB
20130215131209-48800	Caringo, Inc.	04/11/2022	N/A	

Back Install

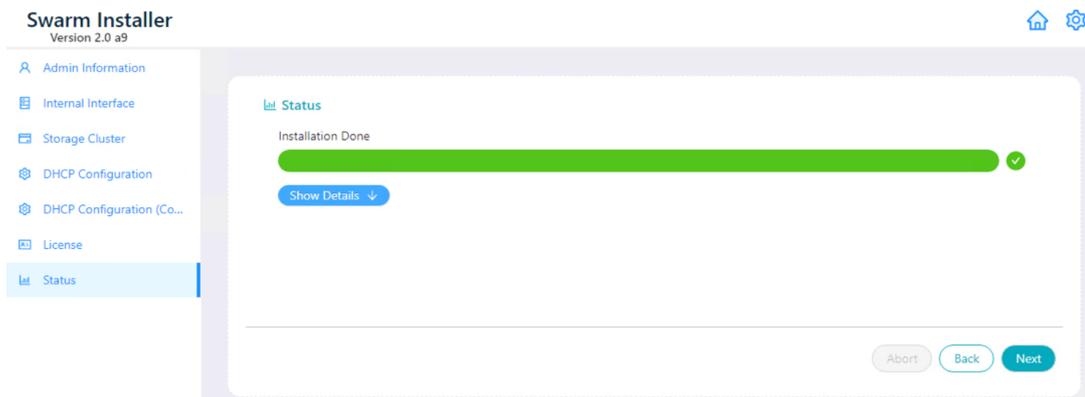
- Click *Install* to proceed with the added license file. It displays a confirmation wizard where click *Yes* to initiate the installation process. Swarm SCS product itself comes with a default license, therefore, this step is optional.

Swarm Installer X

Are you sure want to Install?

No Yes

- Wait for the installation to complete. Click *Show Details* to understand the installation progress or errors. If not, click *Next* to proceed.



The screenshot shows the 'Swarm Installer' interface. On the left is a navigation menu with items: Admin Information, Internal Interface, Storage Cluster, DHCP Configuration, DHCP Configuration (Co...), License, and Status (which is highlighted). The main content area is titled 'Status' and shows 'Installation Done' with a green progress bar and a checkmark. Below the progress bar is a 'Show Details' button. At the bottom right of the main area are 'Abort', 'Back', and 'Next' buttons.

 Next, [\[DRAFT\] Upload YAML File and Start Storage Nodes.](#)

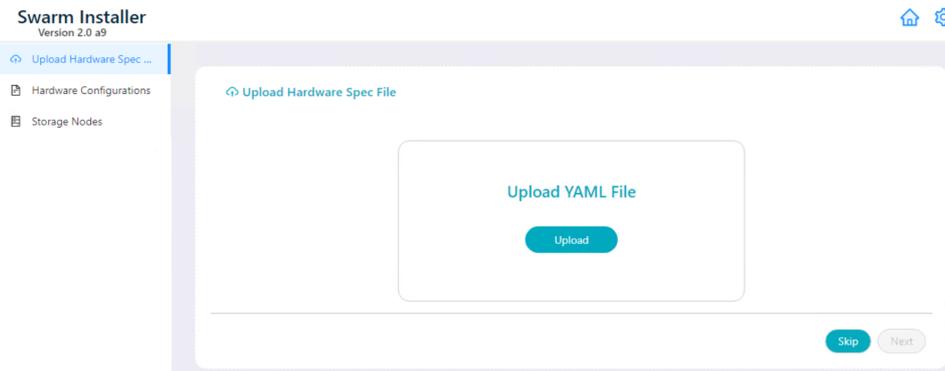
Upload YAML File and Start Storage Nodes

- [Define Hardware Specification](#)
- [Power on Storage Nodes](#)

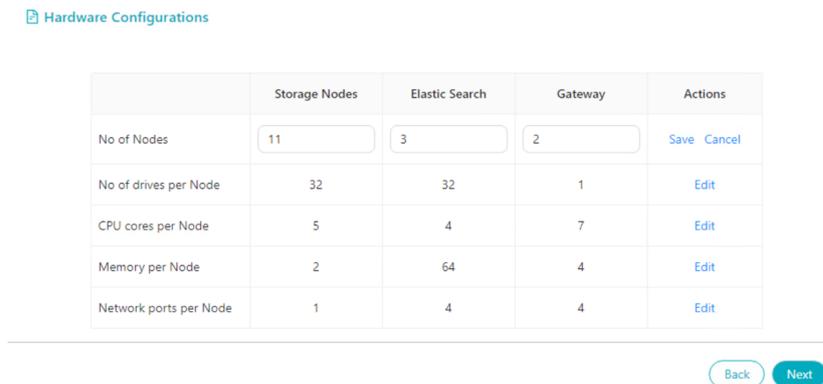
Define Hardware Specification

The YAML file contains hardware specifications and is created using the environment called [Swarm Configurator](#). See [\[DRAFT\] Swarm Configurator](#) for more details about each component used in the configurator.

1. Upload the YAML file exported from Swarm Configurator.



2. Click *Next*. The uploaded file displays the hardware specifications that are editable.



3. Click *Edit* to update any hardware specification if needed. Click *Save* to save the updated details.
4. Click *Next*.

Power on Storage Nodes

1. Swarm Installer displays a wizard to power on storage nodes, click *Ok* to proceed.

Swarm Installer



Power on storage nodes

Cancel

Ok

2. Wait for storage nodes to power on. Once done, details like IP address, hostname, CPU, memory, and maximum storage capacity in TB are displayed for each powered-on storage node. Minimum 3 nodes must be set up and connected to the network to configure the Swarm cluster.

Storage Nodes

IP Address	Hostname	CPU	Memory(GB)	Max storage capacity(TB)
172.30.20.125	chassis-7d11f75ca05783ef983d774e6cd5bb61	1	4	0.066
172.30.20.123	chassis-2fd4cda46dfff3932fd11af45b1e0020	1	4	0.066
172.30.20.124	chassis-3cacde086fb026ee9465a5772e2f9665	1	4	0.066

Skip

Back

Install Elastic Search

3. Click *Install Elasticsearch* to continue.

Next, [\[DRAFT\] Install Elasticsearch](#).

Install Elasticsearch for SCS 2.0

1. On the Elasticsearch Summary page, hardware specifications are prefilled as per data uploaded using the YAML file. These prefilled hardware specifications (number of nodes, CPU cors per node, and memory per node) are editable so the customer can update values as required and click *Next* to proceed.

 Elastic Search Summary

No of Nodes

CPU cores per Node

Memory per Node(GB)

Next

2. Add one or multiple Elasticsearch nodes as required, using *Add* button.
3. Provide the vCenter host information.

Swarm Installer



Vcenter Information

DataCenter

DataStore

ESX Host FQDN

Management Network

Storage Network

Next

Cancel Add

- a. DataCenter - Enter the data center name.
- b. DataStore - Enter the datastore name.

- c. ESX Host FQDN - Enter the ESX hostname.
 - d. Management Network - Public network
 - e. Storage Network - Private network
4. Provide the below Elasticsearch information for installation.

Swarm Installer
✕

Management IP

Static DHCP

Management IP /

Network Gateway

DNS Servers Add

172.30.0.20 x

172.30.0.60 x

Storage IP

Static DHCP

Cancel
Add

- a. Enter the hostname, domain name, and management IP of Elasticsearch.
 - b. If the management IP is static:
 - i. Provide management network IP with network mass subnet and network gateway IP address.
 - ii. Add the number of DNS servers required.
5. Select the storage IP as static or DHCP.
- a. If static IP is selected, provide an IP address and network mask.
6. Click *Add*. It adds Elasticsearch hostname information. The hostname information is editable so a customer can update it if required or can delete it if not required anymore.

Host Information

3/3 [Add](#)

Elastic Search HostName	Management IP	Storage IP	ESX HostName	Actions
es1	172.30.4.80/24	172.30.20.113/24	certlab-esx3.tx.caringo.com	Edit Delete
es2	172.30.4.81/24	172.30.20.114/24	certlab-esx3.tx.caringo.com	Edit Delete
es3	172.30.4.82/24	172.30.20.115/24	certlab-esx3.tx.caringo.com	Edit Delete

[Back](#) [Install](#)

7. Click *Install* to initiate installing Elasticsearch. Wait for Elasticsearch installation to complete.

Status

Run ansible for elasticSearch



[Show Details](#)

[Abort](#) [Back](#) [Install Gateway](#)

i When specified DHCP for the storage/Private network, IP address is assigned.

✓ Next, [Install Gateway for SCS 2.0](#).

Install Gateway for SCS 2.0

Continue on the same installer to install the Gateway.

1. Click *Install Gateway*.
2. On the Gateway Summary page, the number of nodes, CPU cores, and Memory per node, are prefilled based on the YAML file uploaded earlier. These details are editable so a customer can update them if required.

 Gateway Summary

No of Nodes	<input type="text" value="2"/>
CPU cores per Node	<input type="text" value="7"/>
Memory per Node(GB)	<input type="text" value="4"/>

[Next](#)

3. Click *Next*.
4. Click *Add* to add a host node.
5. Provide the VMware host details.

Swarm Installer



Vcenter Information

 DataCenter

 DataStore

 ESX Host FQDN

 Access Network

 Storage Network

- a. DataCenter - Enter the data center name.
 - b. DataStore - Enter the datastore name.
 - c. ESX Host FQDN - Enter the ESX hostname of the vCenter.
 - d. Access Network - Public network
 - e. Storage Network - Private network
6. Provide the Gateway information.

Swarm Installer
✕

Access IP

Static DHCP

Access IP /

Network Gateway

DNS Servers Add

172.30.0.20 x
172.30.0.60 x

Storage IP

Static DHCP

Cancel
Add

- a. Enter the hostname and domain name for Gateway.
 - b. Select an access IP as static or DHCP.
 - i. For a static IP address, provide the access IP with a network mass subnet and network gateway IP address. Add the number of DNS servers required.
 - ii. For DHCP, an IP address is assigned by the SCS.
7. Click *Add*. It adds the host node on the Gateway. Repeat steps 2 to 6 to add more host nodes.

 **Host Information**

1/1 Add

Gateway HostName	Access IP	Storage IP	ESX HostName	Actions
gw1	172.30.4.85/24	172.30.20.121/24	certlab-esx3.tx.caringo.com	Edit Delete

Back
Next

8. Click *Next* to add the authentication method.

9. Select an authentication method as per the authentication type configured on the customer's VM. The drop-down contains [LDAP](#), [Active Directory](#), [SAML](#), or [PAM](#) methods.

Authentication Methods

Authentication Method

- LDAP
- AD
- PAM**
- SAML

[Back](#) [Next](#)

10. Click *Next*.
11. As per the authentication method configured, define a root-level policy allowed for the Swarm cluster.

Policies

User name [Add](#)

Group name [Add](#)

Scope

Action

[Back](#) [Install](#)

12. Provide the list of users and/or groups allowed to access the cluster.
13. Click *Install*. It installs the gateway and launches VM.

Status

Run ansible for gateway

[Show Details](#)

[Abort](#) [Back](#) [Install Telemetry Server](#)

i If PAM authentication is selected, user is created in the local gateway with the username/password provided while configuring SCS node. By default, access policy is defined with the newly created username.

✓ Next, [\[DRAFT\] Install Telemetry Server](#).

Install Telemetry Server

Continue on the same installer to Install the Telemetry.

1. Click *Install Telemetry Server*.
2. On the Telemetry summary page, the number of nodes and CPU/Memory per node are prefilled based on the YAML file uploaded. These details are editable so a customer can update them if required.

Telemetry summary

No of Nodes

CPU cores per Node

Memory per Node(GB)

[Next](#)

3. Click *Next*.
4. Click *Add* to add vCenter host information and Telemetry host nodes.
5. For vCenter host information;

X

Vcenter Information

DataCenter

DataStore

ESX Host FQDN

Management Network

Storage Network

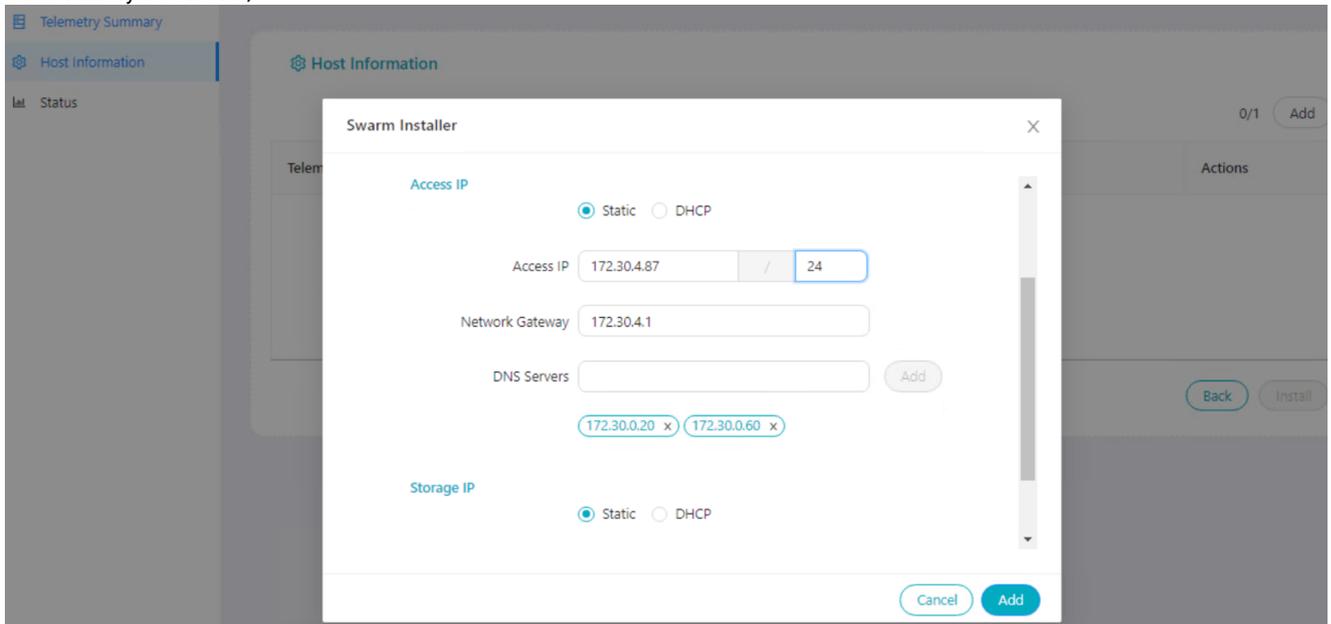
[Next](#)

[Cancel](#) [Add](#)

- a. DataCenter - Enter the data center name.

- b. DataStore - Enter the datastore name.
- c. ESX Host FQDN - Enter the ESX hostname.
- d. Management Network - Public network
- e. Storage Network - Private network
- f. Click *Next*.

6. For Telemetry host nodes;



- a. Provide the hostname and domain name.
- b. Select an access IP as static or DHCP.
 - i. For a static IP address;
 - 1. Provide the access IP with a network mask subnet and network gateway IP address.
 - 2. Add the number of DNS servers required using *Add* button.
 - ii. For DHCP, an IP address is assigned.
- c. Select either Static or DHCP for the storage IP.
 - i. If static IP is selected, provide the IP address and network mask.

7. Click *Add*. It adds the Telemetry host nodes.

 **Host Information**

1/1 Add

Telemetry HostName	Access IP	Storage IP	ESX HostName	Actions
tele1	172.30.4.87/24	172.30.20.127/24	certlab-esx3.tx.caringo.com	Edit Delete

Back Install

The Telemetry host information can be modified after uploading using *Edit*.

- Click *Install* to initiate Telemetry Installation.
- Wait for the Telemetry server installation to complete. When installed, click *Finish*.

Status

Run ansible for telemetry



Show Details ↓

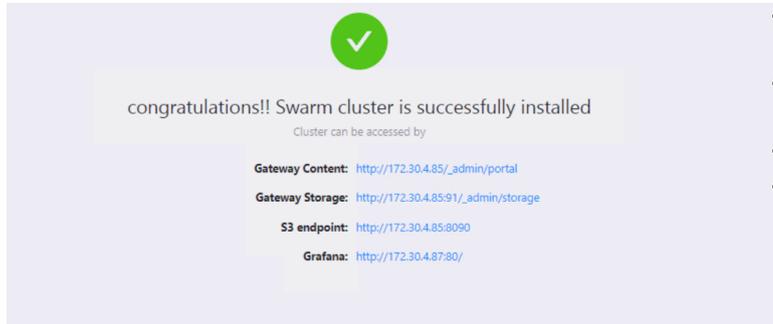
Abort Back Finish

A success message is displayed saying “*congratulation!! Swarm cluster is successfully installed*”.

✔ Next, [\[DRAFT\] Access Links to Dashboard](#).

Access Links to Dashboard

The success screen displays links to access Gateway content and storage, S3 endpoint, and Grafana dashboard.



- **Gateway Content** - http://<ip address-Gateway>/_admin/portal. See [Swarm Content UI](#).
- **Gateway Storage** - http://<ip address-Gateway>:91/_admin/storage. See [Swarm Storage UI](#).
- **S3 endpoint** - <http://<IP address-S3 endpoint>:8090>
- **Grafana** - <http://<ip address-Telemetry server>:80/>

See [Prometheus Node Exporter and Grafana](#) for more information about Grafana dashboard.

Swarm Administration

- [Swarm Storage UI](#)
 - [Managing Chassis and Drives](#)
 - [Managing Feeds](#)
 - [Using Cluster Reports](#)
 - [Swarm UI Essentials](#)
 - [Using Cluster Settings](#)
 - [Health Data to Support](#)
 - [Viewing and Managing the Cluster](#)
 - [Legacy Admin Console \(port 90\)](#)
- [Elasticsearch for Swarm](#)
 - [Snapshot and Restore Search Data](#)
 - [Rebuilding a Search Feed](#)
 - [Monitoring Elasticsearch](#)
 - [Adding Nodes to an ES Cluster](#)
 - [Resetting Elasticsearch](#)
 - [Rolling Restart of Elasticsearch](#)
 - [Uninstalling Elasticsearch](#)
 - [Merging and Renaming ES Clusters](#)
- [Swarm Storage Cluster](#)
 - [Managing Domains](#)
 - [Configuring Swarm Storage](#)
 - [Prometheus Node Exporter and Grafana](#)
 - [Swarm Storage Policies](#)
 - [Managing and Optimizing Feeds](#)
 - [Swarm Concepts](#)
 - [Defining Swarm Admins and Users](#)
 - [Managing Volumes](#)
 - [Using SNMP with Swarm](#)
 - [Troubleshooting Storage](#)
- [Swarm Cluster Services \(SCS\)](#)
 - [SCS Overview](#)
- [SCS Administration](#)
- [Swarm Content Gateway](#)
 - [Managing Dynamic Features](#)
 - [Content Metering](#)
 - [Gateway Operations](#)
 - [Replicating Domains to Other Clusters](#)
 - [Content Gateway Concepts](#)
 - [Upgrading Gateway](#)
 - [Content Gateway Authentication](#)
 - [Gateway Troubleshooting](#)
 - [Object Locking](#)
- [Swarm Content UI](#)
 - [Configuring Domains](#)

- [Setting Identity Management](#)
- [Configuring Buckets](#)
- [Using Virtual Folders](#)
- [Usage Reports](#)
- [Setting Storage Policies](#)
- [Video Clipping for Partial File Restore](#)
- [Configuring Tenants](#)
- [Metadata Encoding](#)
- [Setting Remote Synchronous Write \(RSW\)](#)
- [Editing Names, Metadata, and Versions](#)
- [Using the Content UI](#)
- [Search Collections](#)
- [Setting Permissions](#)
- [Uploading Files](#)
- [Setting Quotas](#)
- [Setting Tokens](#)
- [Content UI Overview](#)
- [Downloading Content](#)
- [Object Locking Content Portal](#)
- [Swarm Hybrid Cloud](#)
- [Bucket Lifecycle Policy](#)
 - [Design & Technical Specifications](#)
 - [Lifecycle Policy Usage & Examples](#)
 - [Client Interfaces](#)
 - [Install & Uninstall Instructions](#)

For information about [Platform Server 10](#), contact DataCore [Support](#).

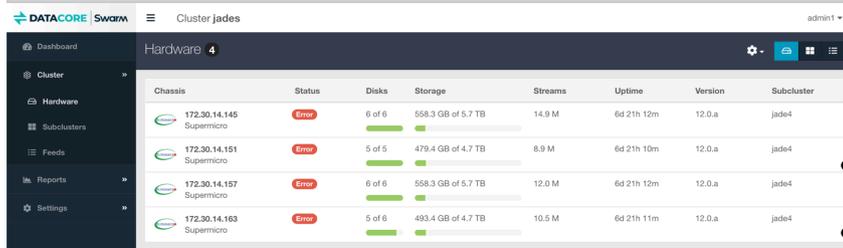
Swarm Storage UI

- [Deprecated](#)
- [Accessing UIs](#)

The Swarm Storage UI (website) presents a comprehensive browser interface for monitoring and controlling your entire Swarm storage implementation.

Deprecated

The [Legacy Admin Console \(port 90\)](#) is still available but is replaced by the Swarm Storage UI. (v10.0)



The website offers your system and storage administrators a unified view of and easy access to the features and settings of Swarm:

- See all cluster chassis and drives, with both real-time and historical status and metrics
- Initiate cluster and chassis-level actions, such as restarting machines or retiring drives
- Create and manage search feeds, and define replication feeds, with optional filtering and SSL encryption
- View and change cluster settings dynamically
- Access event logs and advanced troubleshooting tools
- Identify drive volumes (using the drive light function)
- Monitor the health of the storage cluster, the Elasticsearch cluster, and all search and replication feeds

Accessing UIs

How you access your Swarm websites depends on your configuration:

Site	CSN	Platform
Swarm UI	http://CSN·HOST:CLUSTER_ADMIN·BINDPORT/_admin/storage	http://PLATFORM·HOST:CLUSTER_ADMIN·BINDPORT/_admin/storage
	http://CSN·HOST:91/_admin/storage (default)	http://PLATFORM·HOST:91/_admin/storage (default)
Content UI	http://GATEWAY·IP:SCSP·BINDPORT/_admin/portal	http://GATEWAY·IP:SCSP·BINDPORT/_admin/portal
	http://GATEWAY·IP:80/_admin/portal (default)	http://GATEWAY·IP:80/_admin/portal (default)
Admin Console	http://CSN·HOST:8090/services/storage	http://STORAGE·NODE:90
CSN Console	http://CSN·HOST:8090	n/a

Accessing UIs

- **Bindports** – The `CLUSTER_ADMIN·BINDPORT` and `SCSP·BINDPORT` refer to `bindPort` settings in the [Gateway Configuration](#), in the `[cluster_admin]` and `[scsp]` sections, respectively. You can customize these to support proxies and Docker environments.
- **Storage password** – The Gateway/Service Proxy that is serving the Swarm UI must enable `cluster_admin` and have the Swarm password in `managementPassword`. See [Gateway Configuration](#).
- **User logins** – User logins for the UIs are not Swarm-managed but rather LDAP or PAM, as configured by your IDSYS file, `/etc/caringo/cloudgateway/idsys.json`. See [Gateway Identity System](#).
- **Deprecated** – The functionality of the legacy [CSN Console](#) and [Swarm Admin Console](#) are unified and replaced by the Swarm UI. Both legacy UIs are still available. (v10.0)

- [Managing Chassis and Drives](#)
- [Managing Feeds](#)
- [Using Cluster Reports](#)
- [Swarm UI Essentials](#)
- [Using Cluster Settings](#)
- [Health Data to Support](#)
- [Viewing and Managing the Cluster](#)
- [Legacy Admin Console \(port 90\)](#)

Managing Chassis and Drives

- [Chassis Details](#)
 - [Streams, not objects](#)
 - [Details Tab](#)
 - [Tip](#)
 - [Logs Tab](#)
 - [Message levels](#)
 - [Driver Message Tab](#)
 - [Limited to 1000](#)
 - [Hardware Info Tab](#)
 - [Memory Tab](#)
 - [Best practice](#)
 - [Statistics Tab](#)
 - [Advanced Tab](#)
- [Restarting or Shutting Down a Chassis](#)
- [Retiring a Chassis](#)
 - [Important](#)
 - [Replica protection](#)
- [Retiring a Disk \(Volume\)](#)
- [Identifying a Disk](#)
 - [Note](#)

Chassis Details

Detailed hardware and status information for each chassis (physical or virtual machine) are displayed on the hardware details page.

Chassis	Status	Disks	Storage	Stream index	Streams	Uptime	Version	Subcluster
 172.30.14.145 Supermicro 66.1 GB RAM	Error	6 of 6	151.3 GB of 5.7 TB	0%	2.7 M	4d 11h 42m	10.0.0	jade1

Chassis-wide commands

[Details](#)
[Logs 20](#)
[Driver message](#)
[Hardware info](#)
[Memory](#)
[Statistics](#)
[Advanced](#)

Streams, not objects

Streams are counts of the total number of Swarm-managed data components (such as replicas and segments). Streams are not logical objects (such as video files).

Status states – These are the states reported for hardware in a cluster and how to interpret them:

Status	Nodes / Chassis	Volumes / Disks
ok	Nominal	Nominal
idle	Nominal, but the node is idle	Nominal, but idle
retiring	One or more volumes are offloading streams to the cluster due to retire	Offloading streams to the cluster due to retire
retired	All volumes are retired	Empty of objects and not taking new ones
unavailable		In an error state
error	Errors are reported on the node (hardware or software)	
mounting	One or more volumes are mounting	Mounting at startup/discovery
finalizing	Can appear while the node is rebooting or shutting down, as the node finishes sessions in process	
maintenance	A 3-hour window during an administrative reboot or shutdown where Failed Volume Recovery does not run	

Details Tab

Each detailed row displays a disk name, status, total capacity, amount of used journal space, the largest stream size it contains in MB, Model number, Serial Number, ID, Firmware version, and Encryption status. The **Largest** value displays as 0 if the largest stream on disk is less than 1MB.

Tip

Watch the **Streams** count to track the progress when retiring a disk.

Process Node	Disks	Capacity	Journal	Streams / Largest	Model / Serial Number / ID	Firmware	Encrypted	Disk Lights	Actions
172.30.14.163	/dev/sda	942.7 G	1	1.3 M 0	WDC WD10JFCX-68N WD-WXM1 EA465RWK 91ef95b776eec9361cdc9a1131846f74	0A82	false	On	Retire the disk from here
	/dev/sdb	942.7 G	1	1.2 M 0	WDC WD10JFCX-68N EA45C9X4 5f465d23c437e0d90ca7b6c	0A82	false	Off	

Logs Tab

The **Logs** tab lists the last 10 logged announcements in the cluster as well as the last 10 logged critical alerts. The tab itself includes a count of these messages, and appears red if any are errors:

Severity	Node Process	Date	Message
Info	172.30.14.145	2018-12-18 05:00:56 UTC	Added new feed id=17
Info	172.30.14.145	2018-12-18 05:00:56 UTC	7: ('name': 'navys', 'nodeletes': False, 'lastchanged': 'Tue, 05 Jun 2018 21:44:09 GMT', 'destination': ('insertBatchTimeout': 1, 'host': 'navy1.tx.caringo.com

- Use the **Clear** command to remove log messages which have either been addressed or are not interesting from the display.
- Click the **Log Level** (gear) settings command to view and change the log levels set for this machine.

Hot-swapping disks – Messages display on this tab if a disk is removed or inserted into a running node. This feature, referred to as [Hot Swapping and Plugging Disks](#), allows removal of failed disks for analysis or to add storage capacity to a node at any time.



For example, if adding and then volume, the following messages appears:

```
mounted /dev/sdb, volumeID is 561479FB832DCC526B1D7E1
removed /dev/sdb, volumeID was 561479FB832DCC526B1D7E1
```



Message levels

These messages appear at the **announcement** level. Additional debug level messages appear in the syslog.

Driver Message Tab

dmesg (*driver message*) prints the message buffer of the kernel. These driver messages are useful for diagnosing a Swarm issue when a system panic or error occurs.



Limited to 1000

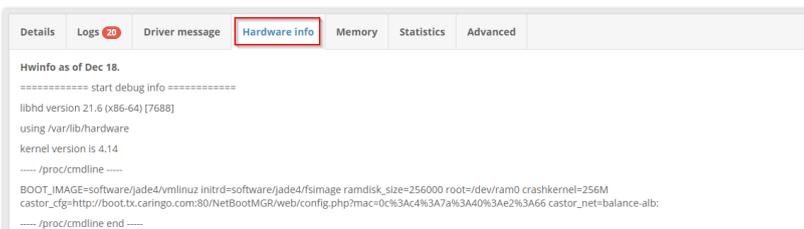
dmesg is a circular buffer; it shows the last 1000 kernel messages.



Hardware Info Tab

hwinfo (*hardware information*) is the Linux hardware detection tool output. This tool probes for the hardware present in the system and displays detailed information about various hardware components in human-readable

format.



Memory Tab

The usage report on the Memory tab provides detailed information to help with troubleshooting insufficient memory.

Each node uses memory to hold an *index* of the objects stored in it. A node stops storing new content until space is freed through deletions if a node runs out of index

space. A full node continues to respond to client read requests for data already present. Each named or alias object requires two index slots. Erasure coding typically requires more memory than replication; exactly how much depends on the encoding.



Best practice

Increase the memory in the node if running out of index slots through normal activity.

Details	Logs 20	Driver message	Hardware info	Memory	Statistics	Advanced
Memory usage as of Dec 18						
Index Memory Reserved	40.0 GB					
Reserve Memory	30.0 MB					
IO Buffer Memory	26.7 GB					
Accounted Memory (High-Water)	541.7 KB					
Accounted Memory (In Use)	536.9 KB					
Accounted Memory (Utilization)	0%					

Statistics Tab

The Statistics tab rolls up a detailed, expandable report combining Health Processor (HP), Communications (cluster network), and Memory usage counts and values, to help with analysis and troubleshooting.

Details	Logs 20	Driver message	Hardware info	Memory	Statistics	Advanced
Statistics as of Dec 18						
<pre> Statistics > Health processor statistics: Object > Communications statistics: Object > Memory: Object accountedMemoryHighwater: 541745 accountedMemoryInUse: 536867 accountedMemoryUtilization: 0 > cache: Object indexSlotsAvailable: 1067413652 indexUtilization: 1 ioBufferMemory: 26668660249 > overlay: Object enabled: true slotsTotal: 1073741824 slotsUsed: 0 status: "only 1 / 4 nodes (32 are needed) 1544723289" totalMem: 40047990374 </pre>						

The health processor runs on each Swarm node to check the status of streams, performing a wide range of actions:

- Sends replica checks to the other nodes and adds or trims replicas based on responses

- Deletes streams requiring deletion according to lifepoints
- Provides a safety net to remove older alias and named stream versions when a newer version is found in the cluster (which can happen when nodes are restored)
- Checks each stream for data corruption via comparison with the stored stream hash
- Moves the stream on disk, if defragmentation is needed
- Verifies the disk index is consistent with the streams found on disk
- Verifies replicas are distributed properly in the cluster

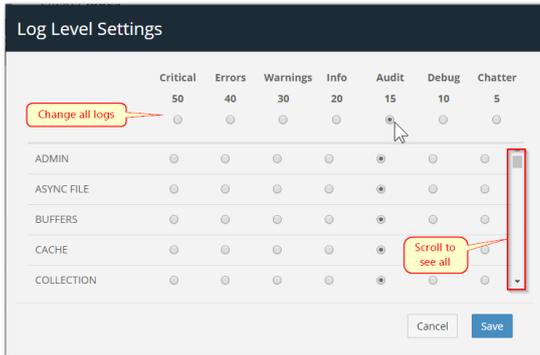
Advanced Tab

The Advanced tab allows dynamically changing machine-level logging levels and also work with Swarm's management API, both through a hands-on HAL browser and a Swagger visualizer.

The Health Data is the raw JSON content of the health report the cluster sends to DataCore Support. See [Health Data to Support](#).

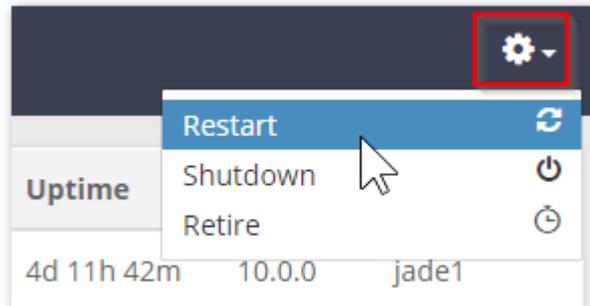
Details	Logs 20	Driver message	Hardware info	Memory	Statistics	Advanced
<div style="display: flex; justify-content: space-between;"> <div style="width: 30%;"> <p>Health Data View the raw JSON</p> <p>View ></p> <p><i>Contents of the health report that Caringo receives</i></p> </div> <div style="width: 30%;"> <p>HAL Browser Explore the API</p> <p>View ></p> </div> <div style="width: 30%;"> <p>Swagger.io UI Visualize the API</p> <p>View ></p> </div> </div> <div style="display: flex; justify-content: space-between; margin-top: 20px;"> <div style="width: 30%;"> <p>Log Level Settings Chassis level debugging</p> <p>View ></p> </div> <div style="width: 30%;"> <p>Software Revision 10.0.rc1 stable</p> </div> </div>						

You can reset the log levels from this tab, as well as from the **Logs** tab:



Restarting or Shutting Down a Chassis

The gear icon at the top of the page allows restarting or shutting down the chassis. A node shut down or rebooted by an Administrator appears with a **Maintenance** state on other nodes in the cluster.



Retiring a Chassis

Retire the chassis when replacing Swarm storage volumes for regular maintenance or to upgrade the cluster chassis with higher

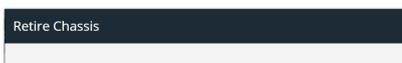
capacity disks. Retiring a chassis copies all objects to other chassis in the cluster, allowing safe removal of the chassis disks without risking any data loss.

Important

Verify the cluster meets the following requirements before retiring a chassis:

- Has enough **capacity** for the objects on the retiring chassis to replicate elsewhere.
- Has enough **remaining nodes** to replicate the objects with one replica on any given node.

Select the **Retire** option under the gear icon at the top of the Chassis Details page to initiate a retire. Choose to perform a minimally disruptive retire limited to the chassis being retired, or an accelerated retire using all nodes in the cluster to replicate objects on the retiring chassis as quickly as possible when initiating a retire. Note: the cluster-wide retire may impact performance as it does put additional load on the cluster.



Replica protection

Retire succeeds if objects can be replicated elsewhere in the cluster. The Retire action does not remove an object until it can guarantee *at least* two replicas exist in the cluster or the existing number of replicas matches the **policy.replicas min** parameter value.

A retiring chassis accepts no new or updated objects. Each chassis volume's state changes to **Retired** and Swarm no longer uses the volume after all objects are copied elsewhere. The volume can be safely removed at this point.

Rate of the retire – Swarm calculates the retire rate over the last hour, which it publishes via SNMP as `retireRatePerHour`. This covers the entire chassis, regardless of how many volumes are being retired.

Canceling the retire – Cancel an in-process retire by selecting the **Cancel Retire** option under the gear icon at the top of the Chassis Details page. Cancel a retire while one or more disks in the chassis have a **Retiring** status.

Retiring a Disk (Volume)

Disk-level retires are useful for targeting bad (slow) disks and for working around having too limited capacity for retires of entire chassis. Check the diagnostic data collected in the logs if a disk retires *automatically* because of I/O errors. (v11.1)

To retire a volume, locate and click the gear icon in the row for the affected disk:

Chassis	Status	Disks	Storage	Stream index	Streams	Uptime	Version	Subcluster
172.30.14.145 Supermicro 66.1 GB RAM	OK	6 of 6	151.3 GB of 5.7 TB	0%	2.7 M	6d 10h 55m	10.0.0	jade1

Process Node	Disks	Capacity	Journal	Streams / Largest	Model / Serial Number / ID	Firmware	Encrypted	Disk Lights	Actions
172.30.14.145	/dev/sda OK	942.7 G	0	448.5 K 0	WDC WD10JFCX-68N WD-WXM1EA49VYSZ 111e5190ae44d183b05d7e263ee424	0A82	false	Off	
	/dev/sdb OK	942.7 G	0	455.5 K 0	WDC WD10JFCX-68N WD-WXM1EA41UV6L 6f24465c5140f3b18f3e169dbfe1e1da	0A82	false	Off	

Choose the speed of retire. The fastest method incurs maximum effort by the cluster to move the content:

Rate of the retire – Swarm generates an announce-level message reporting the overall duration and rate of the retire when Swarm completes a retire task on a disk. (v11.0)

Canceling the retire – Click the gear icon in the row for the affected disk and select the **Cancel retire** command:

Retire Disk

All content will be moved and the disk will permanently be retired.

Background retire (minimal disruption)
 Maximum effort retire (fastest)

Identifying a Disk

It is helpful to enable the LED disk light for the disk when attempting to identify a failed or failing disk. Click on the disk light toggle in the disk's display row to flash the disk light for a specific disk:

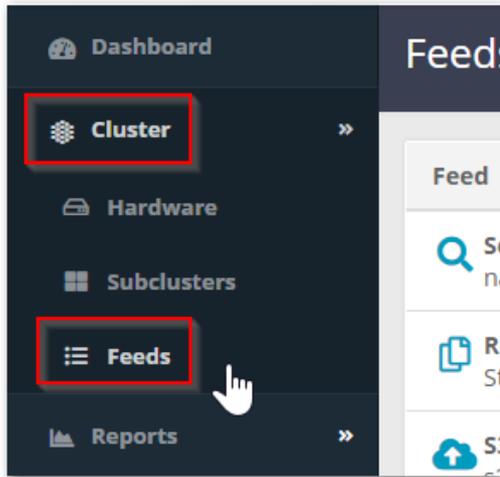
Unavailable	0				WD-WXM1EA42ZRA3 67b9933ff89ef0ff247a93fd4dd6558d			

Note

Disk lights remain lit until manually turned off so return to the Chassis Details page and click the disk light switch to **Off**.

Managing Feeds

Select **Cluster > Feeds** from the global menu to view the **Feeds** page:



The **Feeds** page provides a summary view of all types of feeds are currently configured for your storage cluster, providing you a dashboard for monitoring the entire set. This snapshot includes the status, number of queued events and deletes, processing rate, scope (global or domain-level), and internal feed ID.

Status – These are statuses a feed may show:

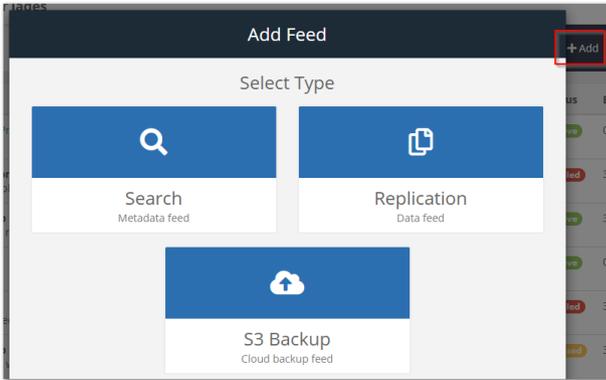
- **Active.** Feed processing is operating normally.
- **Paused.** Feed processing is stopped by user request, and remains so until unpaused.
- **Recovering.** Feed processing is stopped temporarily, due to volume recovery.
- **Stalled.** Feed processing is stopped indefinitely, due to a blocking issue.
- **Configuration error.** Feed cannot operate due to a problem in its configuration.
- **Overlapping feeds.** More than the limit of 8 feeds are defined for the same set of objects.
- **Closed.** Feed is inactive and no longer operational.

Details – Click on any feed row to view or edit the configuration details for that feed.

Feed	Status	Events queue	Deletes queue	Rate	Scope	Target	ID
Replication rep-feed	Active	18,622,266	881	460.7 /hour	Domain	jans222 (POST)	0
Search - Primary navys	Active	0	0	1,460.7 /hour	Global	navys	7
Replication testReplicationFeed	Paused	622,757	635	0 /hour	Global	testReplicationCluster (GET)	9
Replication	Stalled	18,622,680	531	0 /hour	Global	testReplicationCluster	11

New feed – Click **+Add** in the command bar and select from the available feed types to add a feed to your cluster. Feed types are detailed next:

- [Search Feeds](#)
- [S3 Backup Feeds](#)
- [Replication Feeds](#)
- [Replication Feeds over Untrusted Networks](#)



Search Feeds

- [Adding a Search Feed](#)
- [Using Feed Actions](#)
- [Troubleshooting Feeds](#)

Adding a Search Feed

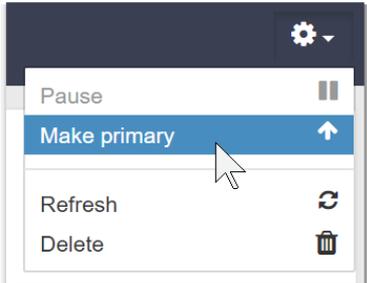
To add a Search feed in the cluster, click the **+Add** button at the top of the **Feeds** page and then add a **Search** feed.

Naming – Swarm applies a naming scheme guaranteeing Elasticsearch index names are always unique, within and across clusters. Swarm creates a new Elasticsearch index and alias for that feed if a second search feed is created through the UI. (v9.0)

Multiples – Swarm allows creation of more than one Search feed facilitating transition from using one Elasticsearch cluster to another. During the transition, continue using the primary feed for queries; the second feed is incomplete until it fully clears its backlog. When the second feed is caught up, transition to it (apply **Make primary** to the second feed) as soon as reasonable for current operations. Delete the original feed once the new primary feed target is verified as working. Having multiple feeds is usually for temporary use only because every feed incurs cluster activity, even when paused.

Important

Restart all Gateway servers to pick up the new feed and update any [SwarmFS Export Configuration](#) if the default search feed is changed (apply **Make primary** to a new feed). No restarts are needed if only a feed definition is updated.



Search Feed

Name

ID (existing feeds)

Batch Timeout (seconds)

Status (existing feeds) Enabled

Target Elasticsearch Cluster

Server host(s) or IP(s)

Server Port

The following table describes the data entry fields in the dialog box.

ID (existing feeds)	Read-only; system-assigned identifier
Status (existing feeds)	Read-only; the current feed processing state.
Primary	Flags the Search feed used for all search queries. Only one feed can be Primary. Set from the Feeds command menu.

Status Active Primary

Name	The name attached to this feed.
-------------	---------------------------------

Batch Size	Defaults to 100. The maximum number of objects sent concurrently to be processed.
Batch Timeout (seconds)	Defaults to 1. The maximum amount of time (in seconds) before a batch is resent to be processed after a timeout.
Search Full Metadata	<p>Enabled - (default) Swarm storage indexes all object metadata, including baseline and custom metadata fields. Disabled - Swarm storage indexes only the baseline metadata fields.</p> <p>See Metadata Field Matching for a list of baseline and custom fields.</p>
Server Host(s) or IP (s)	<p>The IP addresses or server names are resolvable by DNS. Separate with a comma or space if entering more than one. DNS must be configured on both the source and target clusters.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>i Important</p> <p>Refresh the feed to prevent it becoming blocked if the list of ES servers on an active feed is changed.</p> </div>
Server Port	Defaults to 9200. The default port for a host.
Alias (existing feeds)	Read-only; system-assigned name by which Elasticsearch references the Swarm feed.

Adding a feed to an existing index

Use one of the following methods if creating a new search feed pointing to an existing index:

Create alias

1. Create a new ES alias pointing to the existing index:

```
curl -i -X POST <ES-node>:9200/_aliases \
-d '{
  "actions": [
    {
      "add": {
        "index": "EXISTING·INDEX·NAME",
        "alias": "NEW·ALIAS·NAME"
      }
    }
  ]
}'
```

2. Create a new index feed and specify the new ES alias created:

```
curl -i -X POST --anyauth -u admin:ourpwdofchoicehere <swarm-node>:91/api/storage/feed
-d '{
  "actions": [
    {
      "add": {
        "index": "EXISTING·INDEX·NAME",
        "alias": "NEW·ALIAS·NAME"
      }
    }
  ]
}'
```

Remap alias

1. Create an index feed through the UI as usual.
2. Remap the ES alias to the existing index:

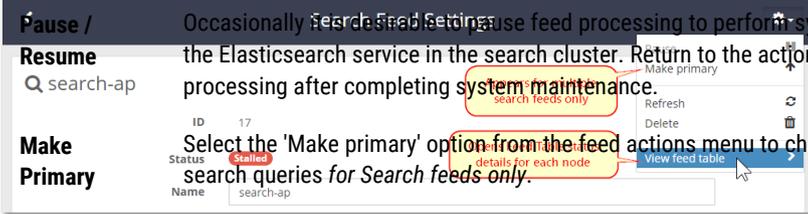
```
curl -i -X POST <ES-node>:9200/_aliases \
-d '{
  "actions": [
    {
      "remove": {
        "index": "NEW·INDEX·NAME",
        "alias": "NEW·ALIAS·NAME"
      }
    },
    {
      "add": {
        "index": "EXISTING·INDEX·NAME",
        "alias": "NEW·ALIAS·NAME"
      }
    }
  ]
}'
```

3. Delete the new index created for the new feed:

```
curl -i -X DELETE <ES-node>:9200/NEW·INDEX·NAME
```

Using Feed Actions

Clicking on an existing search feed in the Feeds list opens its **Feed Settings** page, with the existing settings populated. The gear icon menu at the top right supports multiple feed actions, appropriate to the type of feed:

<p>Pause / Resume</p> <p>Occasionally search feeds use feed processing to perform system maintenance. Pause the search feed before stopping the Elasticsearch service in the search cluster. Return to the action menu and select the Resume action to resume feed processing after completing system maintenance.</p>	 <p>Select the 'Make primary' option from the feed actions menu to change which search feed is the primary feed used for all search queries for Search feeds only.</p>
<p>Refresh</p>	<p>Object data is sent to the feed target in near real-time (NRT) as they are written or updated. Any objects unable to be processed immediately are retried each HP cycle until successful, at which point they are marked as complete and are not resent. Select the Refresh option from the feed action menu, which verifies and rehydrates all previously sent content to the Elasticsearch cluster, if a data loss failure occurs on the remote feed target and a restore from backup cannot be completed. This process takes some time, as it must revisit all objects in the cluster.</p> <p>For search feeds, if an Elasticsearch index for the cluster does not exist, it is created. To recreate an existing index 'fresh' (such as for case-insensitive searching where case-sensitive was previously used), drop the existing index before refreshing the feed.</p>
<p>Delete</p>	<p>When deleting a feed, it frees source cluster resources. To delete a feed, select the Delete option from the feed action menu and verify intention to permanently delete the feed. The deleted feed is removed from the remaining cluster nodes within 60 seconds. Delete the search data previously sent by the feed if desired.</p>
<p>View feed table</p>	<p>Displays the SNMP Repository Dump for the selected node, for feed diagnostics and troubleshooting (see below).</p>

Troubleshooting Feeds

- **Feed diagnostics** – To troubleshoot blocked feed, double-click it to open its settings page, click the gear icon, and select **View feed table**, which displays the SNMP Repository Dump for the selected node. (v2.0)



- Review the **feedPluginState** status to identify the blockage.
- **Idle feeds** – A feed can *appear* to be idle with items still queued for processing. Plan for the fact that feed status reporting is a best-effort snapshot, not a low-latency or guaranteed transaction mechanism.
- **Feed prioritization** – Domain and bucket context objects are prioritized for *all* types of feeds; this improves usability when initiating remote sites.
- **Retries for blocked feeds** – Blocked feeds are retried every 20 minutes, but if the definition for a blocked feed is changed, it triggers an immediate attempt with the new definition, which may clear the blockage. (v10.1)

- **Blocked Search feeds** – Swarm marks the feed with the status **Blocked** and messages report it is missing if Swarm cannot find the Elasticsearch index associated with a search feed. Delete the feed and recreate it with the same settings if the search index is gone.

S3 Backup Feeds

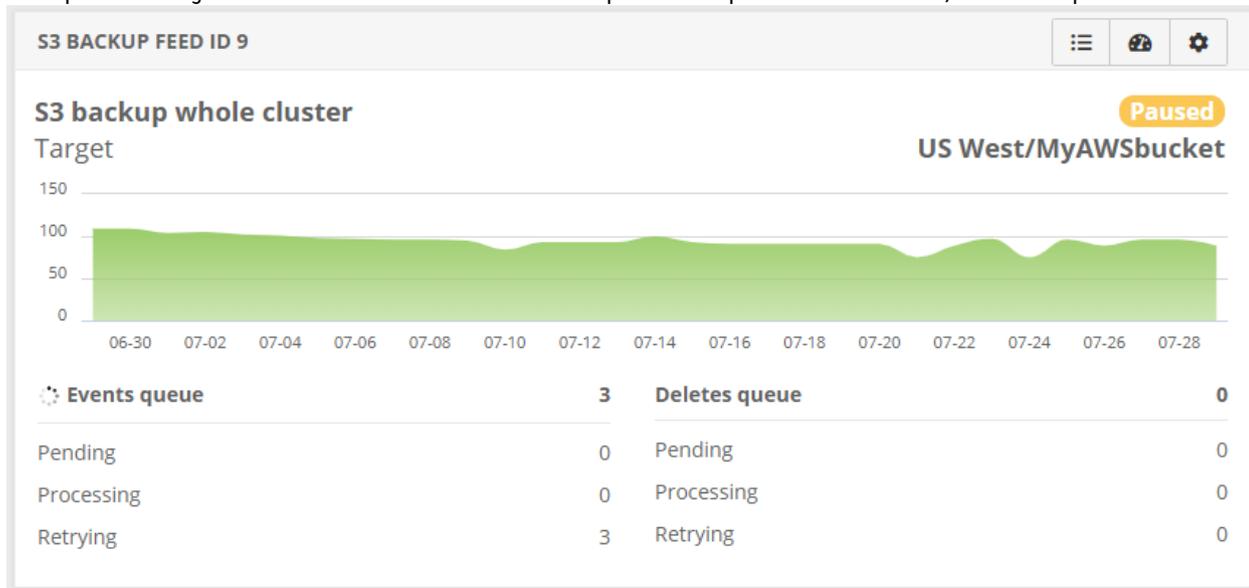
- [Choosing S3 for Disaster Recovery](#)
 - [Important](#)
- [Standard or Cold Storage?](#)
- [Setting up the S3 Bucket](#)
 - [Note](#)
- [Configuring the S3 Backup Feed](#)
 - [Caution](#)
- [Rotating the S3 Access Key](#)
- [Using Feed Actions](#)
- [Troubleshooting Feeds](#)

Choosing S3 for Disaster Recovery

In addition to on-premises Swarm storage, an organization may want to take advantage of public cloud services for off-premises disaster recovery (DR) storage. With the S3 protocol support, Swarm provides choices of many public cloud providers including AWS S3, AWS S3 Glacier, and Wasabi.

The security of knowing backups are continuous, have minimal latency, and require little intervention and monitoring by implementing an S3 backup feed from Swarm. Using Swarm's feed mechanism for backup leverages numerous existing strengths: the long-term iteration over objects in the cluster, proven method for tracking work as it is performed, and support for TLS network encryption and forward proxies. Using the parallelism of the entire Swarm cluster makes best use of network bandwidth, while sending the backups through an optional forward proxy allows implementing bandwidth throttling if needed.

Back up – S3 Backup is an integral part of the operating Swarm cluster. In the Swarm UI, create a new feed of type S3 Backup, provide credentials and information about the network path to the service. After the feed is started, progress can be monitored and warning of blockages and particular object failures can be sent, as with any other feed. The S3 Backup feed honors the [versioning settings in a cluster](#), as enabled, disabled, or suspended throughout the domains and buckets. While multiple S3 Backup feeds can be created, each one requires a dedicated target bucket.



Clean up – No action is needed to keep the backup current and trimmed. When disabling Swarm versioning on buckets or domains, delete buckets or domains, or have [object lifepoints expire](#), the Swarm feeds mechanism processes the expired content as deleted, allowing the S3 Backup feed to clear them from the S3 bucket. Throughout content additions and deletions, the total number objects in the S3 bucket always approximates *twice* the number of logical objects backing up from the source cluster (because AWS functionality requires there to be one for the object's content and another for metadata).

Restore – The Restore tool runs outside of Swarm, using a command-line interface for executing the data and restoration tasks. Restore what is needed: either the entire cluster, or portions. Swarm supports bulk restores at the granularity of cluster, domain, or bucket, as well as more surgical restores of a few objects. Multiple copies can be run to achieve a faster, parallel recovery. See the [S3 Backup Restore Tool](#).



Important

Objects in the S3 backup bucket are wholly dedicated to DR for Swarm and are *not* for general use by owners of the account where the bucket resides. Swarm uses a very specific naming convention within the backup bucket to provide 100% fidelity for object restoration. No external processes other than Swarm should manipulate the content within this bucket.

Standard or Cold Storage?

Swarm 12 supports the [AWS S3 storage classes](#) for standard buckets and those using S3 Glacier and S3 Glacier Deep Archive. For this discussion, *cold storage* refers to S3 Glacier and S3 Glacier Deep Archive, and *standard storage* refers to the traditional S3 storage classes.

Refer to the documentation for the public cloud provider, and consider these points when choosing among the AWS S3 storage classes:

- Cold storage offers the lowest monthly prices per byte stored compared to the standard storage classes.
- Standard storage classes have low-latency retrieval times, which can allow a Swarm Restore to complete in a single run.
- Cold storage has longer retrieval latency, as much as 12-48 hours for S3 Glacier Deep Archive, to pull content from archival storage. Depending upon how a restore is performed, the Swarm Restore tool may need to be run multiple times over several hours to complete a restoration.
- Cold storage incurs additional charges for egress and API requests to access the backup, so it is best suited to low-touch use cases.
- S3 Glacier Deep Archive rounds up small objects, so the overall footprint being charged may be larger because of Swarm's use of metadata objects.

Public storage pricing is competitive, and services such as [Wasabi Hot Cloud Storage](#) may compare favorably with AWS cold storage, especially when considering egress and API charges.

Setting up the S3 Bucket

To implement an S3 backup feed, first complete a one-time set up of the S3 side: set up an account with an S3 cloud service provider and then create an S3 bucket dedicated to backing up *this cluster*.



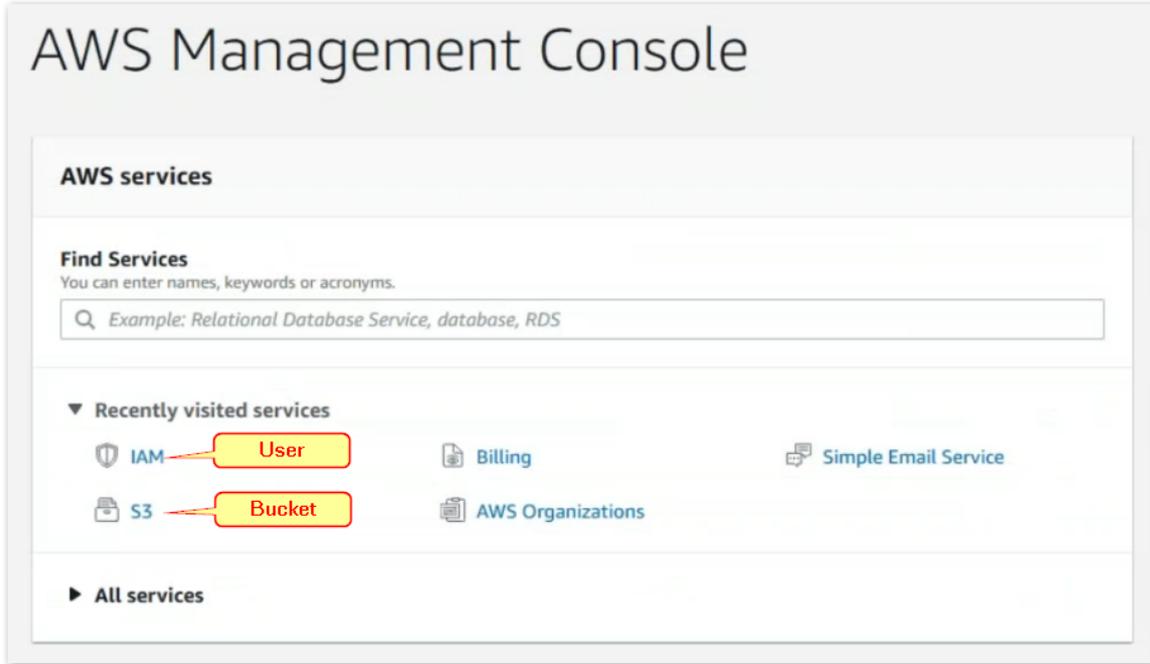
Note

Swarm must be granted access to the target S3 bucket and provide login credentials as part of the S3 backup feed configuration. Neither the S3 Backup feed nor the [S3 Backup Restore Tool](#) administers the S3 credentials or create any target S3 buckets.

While these instruction steps are for AWS S3 (see also [S3 Backup Feeds to Wasabi](#)), S3-based public cloud providers have a similar setup process:

1. **Service** – Sign up for Amazon S3 if needed.

- a. Navigate to aws.amazon.com/s3 and choose **Get started with Amazon S3**.
- b. Follow the on-screen instructions.
- c. AWS notifies by email when the account is active and ready to use.
- d. Note: **S3** is accessed for the new bucket but the separate **IAM** service for the new user:



2. **Bucket**

– Create a bucket dedicated to backing up the Swarm cluster.

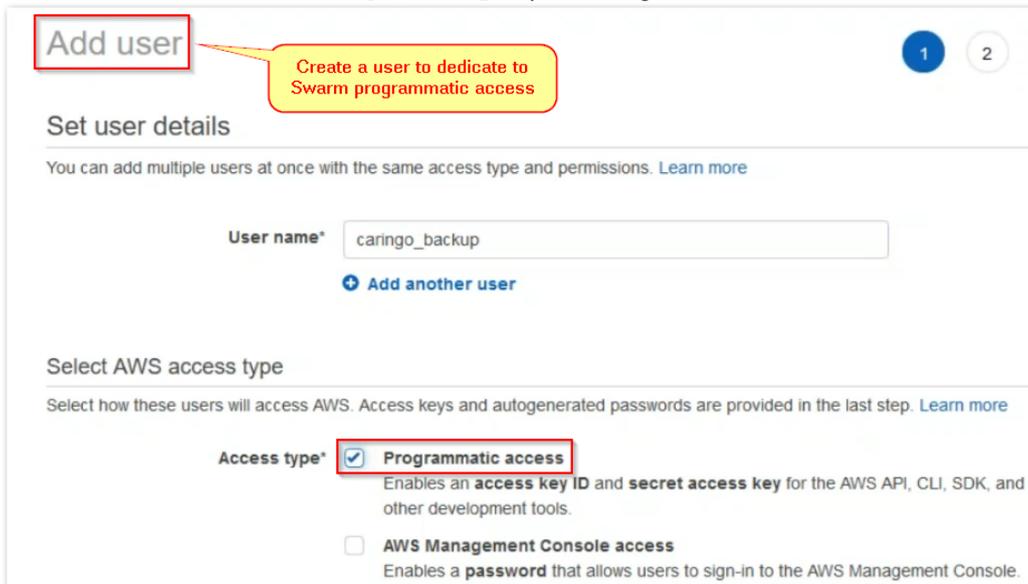
- a. Sign in and open the S3 console: console.aws.amazon.com/s3
- b. Choose **Create bucket**. (See S3 documentation: [Creating a Bucket](#).)
- c. On tab **1 - Name and region**, make the initial entries:
 - i. For **Bucket name**, enter a DNS-compliant name for the new bucket. This cannot be changed later, so choose well:
 1. The name must be unique across all existing bucket names in Amazon S3.
 2. The name must be a valid DNS name, containing lowercase letters and numbers (and internal periods, hyphens, underscores), between 3 and 64 characters. (See S3 documentation: [Rules for Bucket Naming](#).)
Tip: For easier identification, incorporate the name of the Swarm cluster that this bucket is dedicated to backing up.
 - ii. For **Region**, choose the one that is appropriate for business needs. (See S3 documentation: [Regions and Endpoints](#).)
- d. On tab **2 - Configure options**, take the defaults. (See S3 documentation: [Creating a Bucket](#), step 4.)
Best practice: Do not enable versioning or any other optional features, unless it is required for the organization.
- e. On tab **3 - Set permissions**, take the default to select **Block all public access**; now the bucket owner account has full access.
Best practice: Do not use the bucket owner account to provide Swarm's access to the bucket; instead, create a new, separate IAM user that holds the credentials to share with Swarm.
- f. Choose **Create**, and record the fully qualified bucket name (such as "arn:aws:s3:::example.cluster1.backup") for use later, in policies.
- g. Record these values for configuring the S3 Backup feed in Swarm:
 - **Bucket Name**
 - **Region**

3. **User** – Create a programmatic (non-human) user dedicated to Swarm access.

- a. On the Amazon S3 console, select the service **IAM** (Identity and Access Management), click **Users**.



- b. Add a dedicated user, such as `caringo_backup`, to provide **Programmatic access** for Swarm.



- c. The IAM console generates an access key (an access key ID + secret access key), which must be recorded immediately. (See S3 documentation: [Managing Access Keys for IAM Users](#) and [Understanding and Getting Your Security Credentials](#).)

- *This is the sole opportunity to view or download the secret access key, so save it in a secure place.*

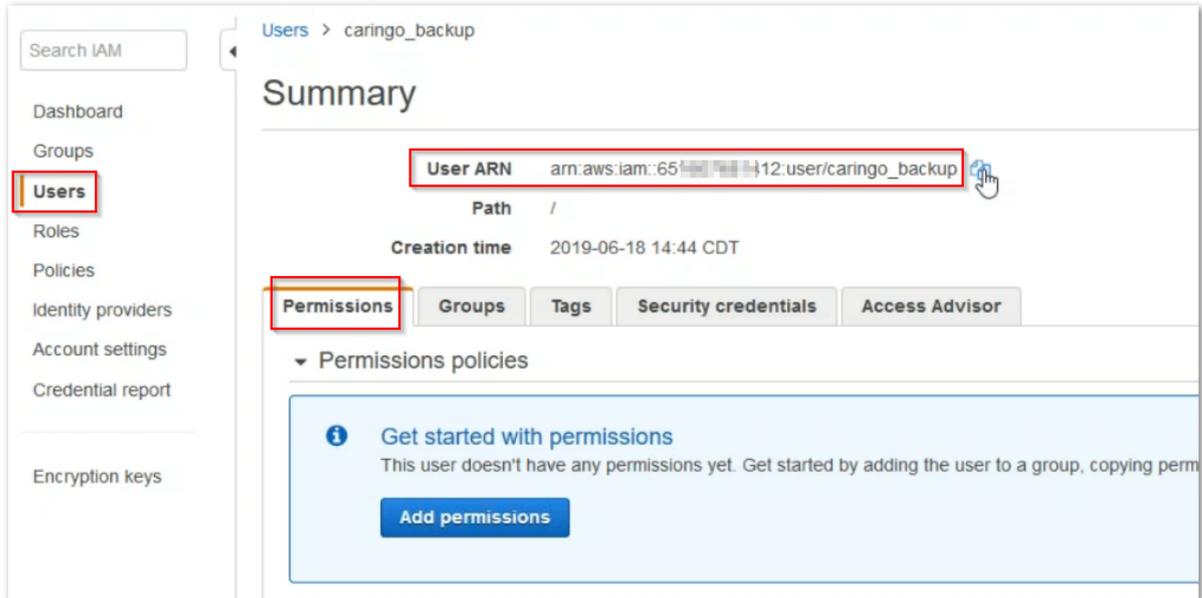
- d. Record the fully qualified user (such as "arn:aws:iam::123456789012:user/caringo_backup") for use later, in policies.

- e. Record these values for configuring the S3 Backup feed in Swarm:

- **Access Key ID**
- **Secret Access Key**

4. **Policies** – Create policies on *both* the user and the bucket so the programmatic user has exclusive rights to the S3 bucket. Use the policy generators provided or enter edited versions of the examples below.

- a. Create an **IAM policy** for this user, allowing it all S3 actions on the backup bucket, which need to be specified as a fully qualified Resource (recorded above), starting with `arn:aws:s3:::`



IAM policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::example.cluster1.backup"
    }
  ]
}
```

- b. Create a matching **bucket policy** to grant access to the dedicated backup user, which need to be specified as a fully qualified **Principal**, which is the User ARN (recorded above) starting with `arn:aws:iam::` (See S3 [Using Bucket Policies.](#)) Using the Policy Generator, allow *all* S3 actions for the bucket, using the full ARN name:

Step 1: Select Policy Type

A Policy is a container for permissions. The different types of policies you can create are an IAM Policy, an S3 Bucket Policy, an SNS Topic Policy, a VPC Endpoint Policy, and an SQS Queue Policy.

Select Type of Policy

Step 2: Add Statement(s)

A statement is the formal description of a single permission. See a [description of elements](#) that you can use in statements.

Effect Allow Deny

Principal

Use a comma to separate multiple values.

AWS Service All Services (**)

Use multiple statements to add permissions for more than one service.

Actions All Actions (**)

Amazon Resource Name (ARN)

ARN should follow the following format: arn:aws:s3:::<bucket_name>/<key_name>. Use a comma to separate multiple values.

[Add Conditions \(Optional\)](#)

Step 3: Generate Policy

Bucket policy

```
{
  "Id": "Policy1560809845679",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1560809828003",
      "Action": "s3:*",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::example.cluster1.backup",
      "Principal": {
        "AWS": [
          "arn:aws:iam::123456789012:user/caringo_backup"
        ]
      }
    }
  ]
}
```

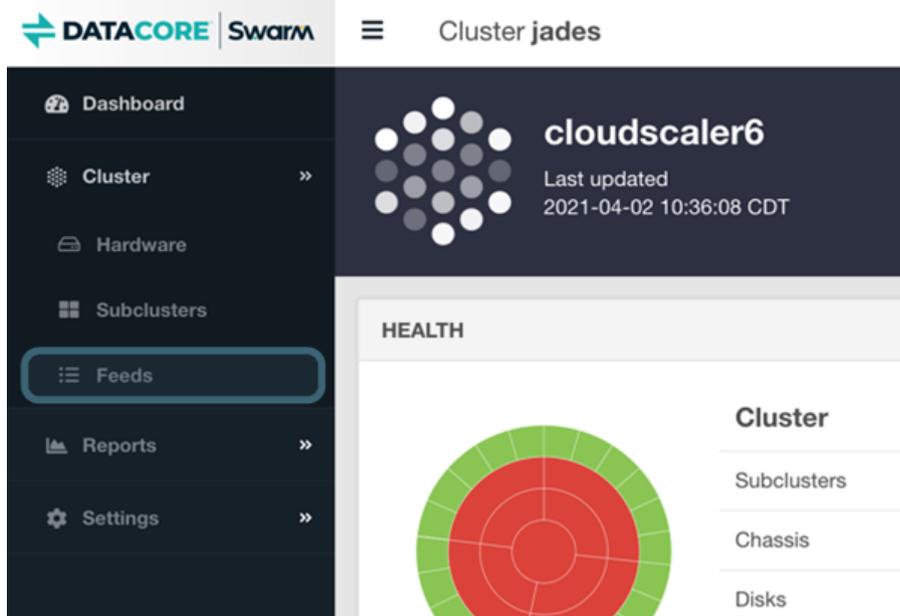
- 5. *Best practice for security:* After implementing the S3 Backup feed in Swarm, write a script to automate rotation of the S3 secret access key on a regular basis, including updating in the S3 Backup feed definition in Swarm (using the management API call, given in *Rotating the S3 Access Key*, below).

Configuring the S3 Backup Feed

The S3 Backup Feed option is available in Swarm 11 and higher, and it may be used immediately after upgrading Swarm Storage. (v11.0)

In addition to Swarm's other feed types, **Search** and **Replication**, a dedicated **S3 Backup** feed can be created. It resembles a Replication feed, but it requires an S3 bucket as the destination and has defaults appropriate for use with a cloud service.

1. Navigate to the **Feeds** page.



2. Select **+ Add** at the top right.



3. Choose **the S3 Backup** feed type:

An S3 Backup feed has these parameters:

ID (existing feeds)	Read-only; system-assigned Identifier						
Status (existing feeds)	<p>Read-only; the current feed processing state. The state can be:</p> <table border="1"> <thead> <tr> <th>Scope</th> <th>Target</th> <th>ID</th> </tr> </thead> <tbody> <tr> <td>Global</td> <td>navys</td> <td>7</td> </tr> </tbody> </table> <ul style="list-style-type: none"> • Active. Default state when operating normally. • Recovering. Temporarily paused due to volume recovery. • Paused. Paused by user request. • Blocked. Processing blocked due to a transient condition. • Configuration error. Feed is unable to operate due to incorrect configuration • Overlapping feeds. More than the limit of 8 feeds are defined for the same set of objects. • Closed. Feed is inactive and no longer operational. 	Scope	Target	ID	Global	navys	7
Scope	Target	ID					
Global	navys	7					
Name	The name attached to this backup feed.						

<p>Scope</p>	<p>The scope filter selected for the backup feed. Backup includes objects within the scope indicated here. If the scope includes a context where Swarm object versioning is (or was) generating historical versions, those versions are backed up as well.</p> <div data-bbox="215 210 1053 798" style="border: 1px solid #ccc; padding: 5px;"> <p style="text-align: center;">Add Feed</p> <p style="text-align: center;">Select Type</p> <ul style="list-style-type: none"> • Entire source cluster (global) – To replicate <i>all</i> objects in the source cluster, leave the default selection of Entire source cluster (global) • Only objects in select domain(s) – To replicate the objects in one or more domains, select the 'Only objects in select domain(s)' option. In the text box that appears, enter one or more domains: <ul style="list-style-type: none"> • To replicate the objects within a <i>specific domain</i>, enter that domain. • To replicate the objects within <i>multiple domains</i>, enter those domains separated by commas and/or use pattern matching. • To exclude domains from replication, enter them. <p>The field value allows pattern matching with the Python regular expression (RE) syntax so multiple domain names can be matched. The exception to the RE matching is that the "{m,n}" repetitions qualifier may not be used.</p> <p>An example domain list value using RE is: <code>.*\example\.com</code></p> <p>This matches both of these domains: accounting.example.com , engineering.example.com.</p> </div> <div data-bbox="311 829 1452 1165" style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Scope <input type="radio"/> Entire source cluster (global) <input checked="" type="radio"/> Only objects in select domain(s)</p> <p>Include domains</p> <p><input type="text" value=".*\example\.com"/></p> <p>Exclude domains</p> <p><input type="text" value="private.example.com"/></p> <p><input type="checkbox"/> Include objects without a domain</p> </div> <p>– To replicate any unnamed objects that are not tenanted in any domain, enable the option.</p>
<p>Target S3 Provider</p>	<p>The configuration for the S3 bucket.</p> <div data-bbox="311 1333 1484 1596" style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Caution</p> <p>Although it is possible to specify another Swarm cluster (via Content Gateway S3) for the S3 backup, it is risky if there is any chance of it replicating back to the <i>source</i> cluster: both clusters can fill to capacity with backups of backups. Best practice is to use a regular Replication feed, which has the mechanisms needed for mirroring clusters safely.</p> </div>

Host	<p>From the S3 configuration, the host name of the S3 service. An IP address cannot be used here because the host name itself becomes the Host header in the feed operation, which is required for communication to S3 services.</p> <p><i>Important:</i> Add the bucket name as the prefix to the host name (<code>mybackup.s3.aws.com</code>). This prefix must match the bucket name exactly, including case. This supports the new AWS bucket-in-host request style. If the bucket is not defined here, Swarm uses the legacy bucket-in-path (<code>s3.aws.com/mybackup</code>) request style. (v12.0)</p> <ul style="list-style-type: none"> • Amazon AWS: Existing feeds are not required to change to this format immediately, but new ones should, as bucket-in-path is unsupported in the future. • Other S3 provider: Verify the provider supports the bucket-in-host request style, where the bucket is part of the FQDN; if not, use bucket-in-path.
Port	<p>The port to use for the S3 service, which defaults to 443 (for HTTPS) or else 80 (HTTP), if Require trusted SSL is disabled, below. If the port is customized, the value no longer updates based on changes to the SSL setting.</p>
Region	<p>From the S3 configuration, the destination S3 bucket's region.</p> <p><i>Note:</i> Changing this value triggers a restart of the feed.</p>
Bucket	<p>From the S3 configuration, the destination S3 bucket name. This bucket must be dedicated to one source cluster. Complete this field regardless of whether the Host includes the bucket name as a prefix.</p> <p><i>Note:</i> Changing this value triggers a restart of the feed.</p>
Access key ID and secret key	<p>From the S3 configuration, the S3 access key ID and S3 secret access key to use. (See S3 documentation: Understanding and Getting Your Security Credentials.)</p> <p>Swarm protects the secret key as a secure field, and hides it. Updating the key does not trigger a restart of the feed, so keys may be updated as frequently as the security policies require.</p>
SSL Server	<p>For production usage, select Require trusted SSL.</p> <p><i>Recommended:</i> To keep bandwidth usage by the S3 Backup feed in check, select the option to use a Local Cluster Forward Proxy and configure one for that purpose. The Forward Proxy Host (hostname or IP address) and Port are required.</p>
Threads	<p>The default backup speed (6 simultaneous threads) is optimal for maintaining an existing S3 backup.</p> <p>For a faster initial backup, increase the threads temporarily, but monitor bandwidth and cluster performance, as boosting the speed stresses internet bandwidth.</p>

Rotating the S3 Access Key

It is a DevOps best-practice to routinely change cloud access credentials and to automate this S3 access key rotation for the S3 Backup feed.

1. Through the public cloud provider, create a new S3 access key and grant the correct permissions for the target S3 bucket.
2. Using Swarm's management API, update the access credentials for the existing S3 backup feed.
3. Expire/remove the old S3 access key upon confirming successful feed operations with the new credentials.

The following command template demonstrates how to use the Swarm management API to update the access credentials for an existing S3 backup feed:

```
curl -X PATCH --header 'Content-Type: application/json' -u <admin>:<password> -d '{
  "op": "replace", "path": "destination/accessKeyId", "value": "<newAccessKeyId>",
  "op": "replace", "path": "destination/secretAccessKey", "value": "<newSecretAccessKey>"}' \
'http://<nodeIP>/api/storage/s3backupfeeds/<s3feedid>'
```

- `<admin>` – The Swarm administrative user name, which is usually `admin`.
- `<password>` – The Swarm administrative password, required for all management API calls that perform actions.
- `<newAccessKeyID>` – The new access key ID for the target S3 bucket.
- `<newSecretAccessKey>` – The new secret access key for the target S3 bucket.
- `<nodeIP>` – The IP address of any Swarm node in the cluster.
- `<s3feedid>` – The small integer feed ID that is associated with the S3 Backup feed. It appears as the feed's **ID** field in the Swarm UI.

Using Feed Actions

Clicking on an existing feed in the Feeds list opens the **Feed Settings** page, with the existing settings populated. The Actions (gear) icon menu at the top right supports multiple feed actions, appropriate to the type of feed:

Pause / Resume	Feed processing may occasionally need to be paused to perform system maintenance. Pause the search feed before stopping the Elasticsearch service in the search cluster when upgrading an Elasticsearch cluster. Return to the action menu and select the Resume action to resume feed processing after completing system maintenance.
Refresh	Object data is sent to the feed target in near real-time (NRT) as they are written or updated. Any objects unable to be processed immediately are retried each HP cycle until successful, at which point they are marked as complete and are not resent. Select the Refresh option from the feed action menu, which verifies and rehydrates all previously sent content to a remote cluster if a data loss failure occurs on the remote feed target and restoration from backup is not possible. This process takes some time, as it must revisit all objects in the cluster.
Delete	When a feed is deleted, it frees source cluster resources. This process does not affect the objects previously pushed to the remote target. Select the Delete option from the feed action menu and verify the intention to permanently delete the feed. The deleted feed is removed from the remaining cluster nodes within 60 seconds.
View feed table	Displays the SNMP Repository Dump for the selected node, for feed diagnostics (see below).

Troubleshooting Feeds

- **Feed diagnostics** – Double-click a blocked feed to open the settings page, click the gear icon, and select **View feed table**, which displays the SNMP Repository Dump for the selected node to troubleshoot. (v2.0)



Review the **feedPluginState** status to identify the blockage.

Example: `feedPluginState blocked: Destination cluster onyx1 reports invalid request: Castor-System-Cluster value must refer to a remote cluster on RETRIEVE request`

- **Idle feeds** – A feed can *appear* to be idle with items still queued for processing. Plan for the fact that feed status reporting is a best-effort snapshot, not a low-latency or guaranteed transaction mechanism.
- **Feed prioritization** – Domain and bucket context objects are prioritized for *all* types of feeds; this improves usability when remote sites are initiated.
- **Retries for blocked feeds** – Blocked feeds are retried every 20 minutes, but if the definition for a blocked feed is changed, it triggers an immediate attempt with the new definition, which may clear the blockage. (v10.1)
- [S3 Backup Feeds to IBM Cloud Object Storage](#)
- [S3 Backup Feeds to RStor](#)
- [S3 Backup Feeds to Seagate Lyve](#)
- [S3 Backup Feeds to Wasabi](#)
- [S3 Backup Restore Tool](#)

S3 Backup Feeds to Wasabi

- [Dedicated Backup](#)
- [Setting up the S3 Bucket](#)
 - [Note](#)
- [Configuring the S3 Backup Feed](#)

Dedicated Backup

Objects in the S3 backup bucket are wholly dedicated to disaster recovery for Swarm and are not for general use by owners of the account where the bucket resides. Consider this feature a restricted form of S3, with constraints on the bucket's namespace that support Swarm's ability to backup and restore. Do not expect the namespace to be end-user friendly.

Swarm S3 backups to Wasabi targets are verified. Complete a one-time set up of the destination to implement an S3 backup feed: set up an account with Wasabi and then create an S3 bucket dedicated to backing up *this cluster*.

Setting up the S3 Bucket

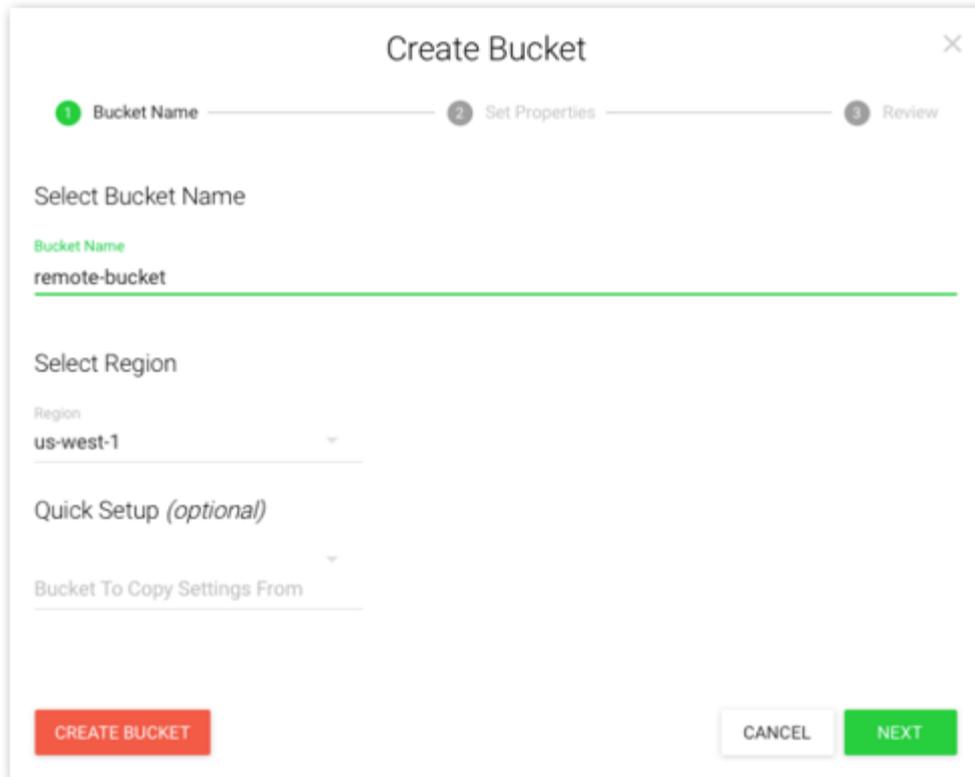
Note

Swarm has the Wasabi access granted it as part of this configuration. Neither the S3 Backup feed nor the [S3 Backup Restore Tool](#) administers S3 credentials or create any S3 buckets in Wasabi.

See [Wasabi Support](#) for assistance.

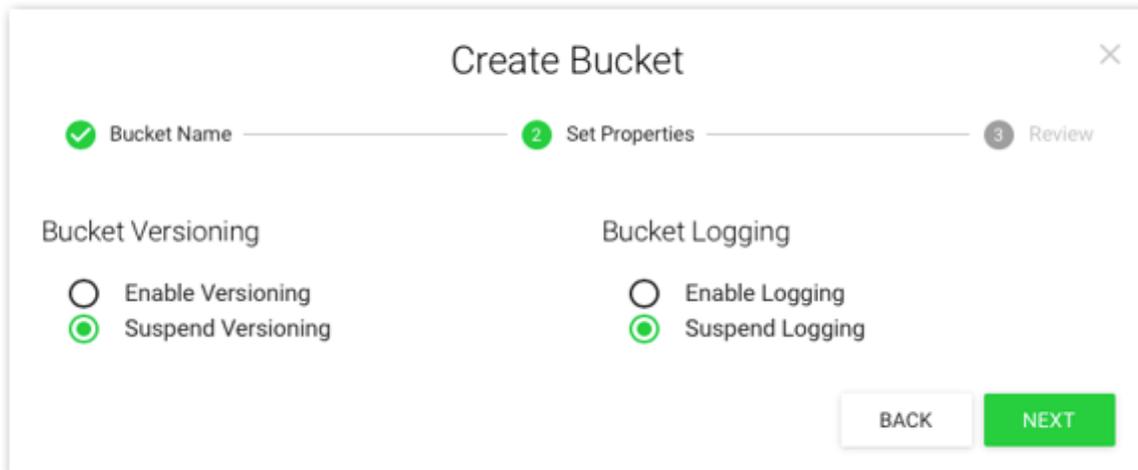
These instructions are for [Wasabi](#) cloud storage, but any Internet-based S3 service has similar functionality:

1. **Service** – Sign up for Wasabi if needed.
 - a. Navigate to [Wasabi's Sign Up page](#), fill out all fields and select **Start Your Free Trial**.
 - b. Follow the on-screen instructions.
 - c. Log in to the Wasabi console once setup is finished.
2. **Bucket** – Create a bucket dedicated to backing up the Swarm cluster.
 - a. Sign in to the Wasabi console: console.wasabisys.com.
 - b. Choose **Create bucket**.
 - c. Make entries for **Name and region**:
 - i. Enter a DNS compliant name for the new bucket for **Bucket name**. Buckets cannot be renamed, so choose wisely:
 1. The name must be unique across all existing bucket names in Wasabi S3.
 2. The name must be a valid DNS name, containing lowercase letters and numbers (and internal periods, hyphens, underscores), between 3 and 64 characters.
(See S3 documentation: [Rules for Bucket Naming](#).)
Tip: Incorporate the name of the Swarm cluster this bucket is dedicated to backing up for easier identification.
 - ii. Choose a **Region** appropriate for business needs.



d. Take the defaults for **Configure options**. *Best practice:* Do not enable versioning or logging unless it is required for the organization.

e. Choose **Create**, and record the fully qualified bucket name (such as "arn:aws:s3:::remote-bucket") for use later, in



f. Record these values for configuring the S3 Backup feed in Swarm:

- **Bucket Name**
- **Region**

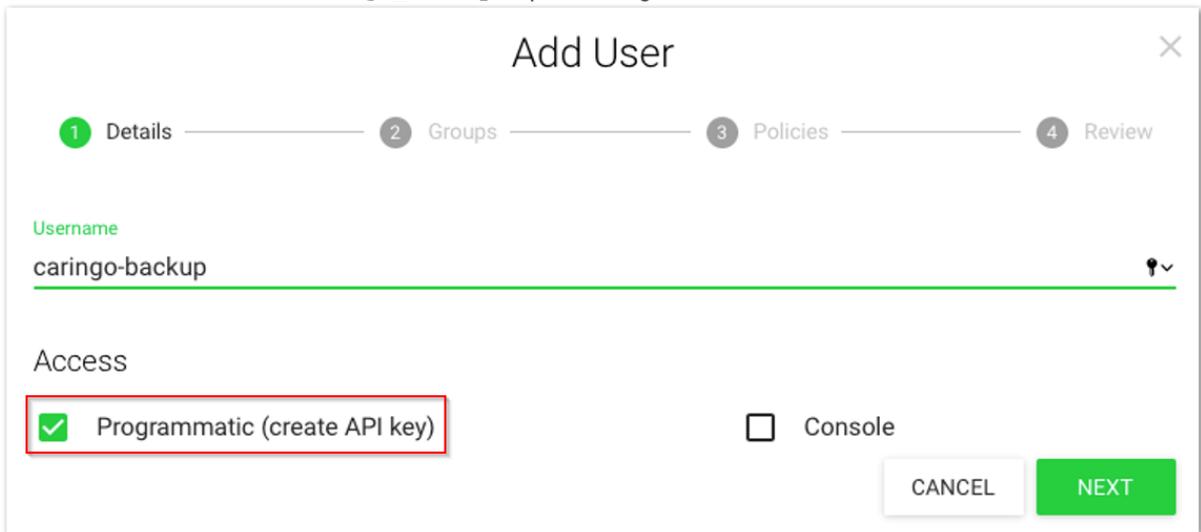
3. **User** – Create a programmatic (non-human/console) user that dedicated to Swarm access.

police

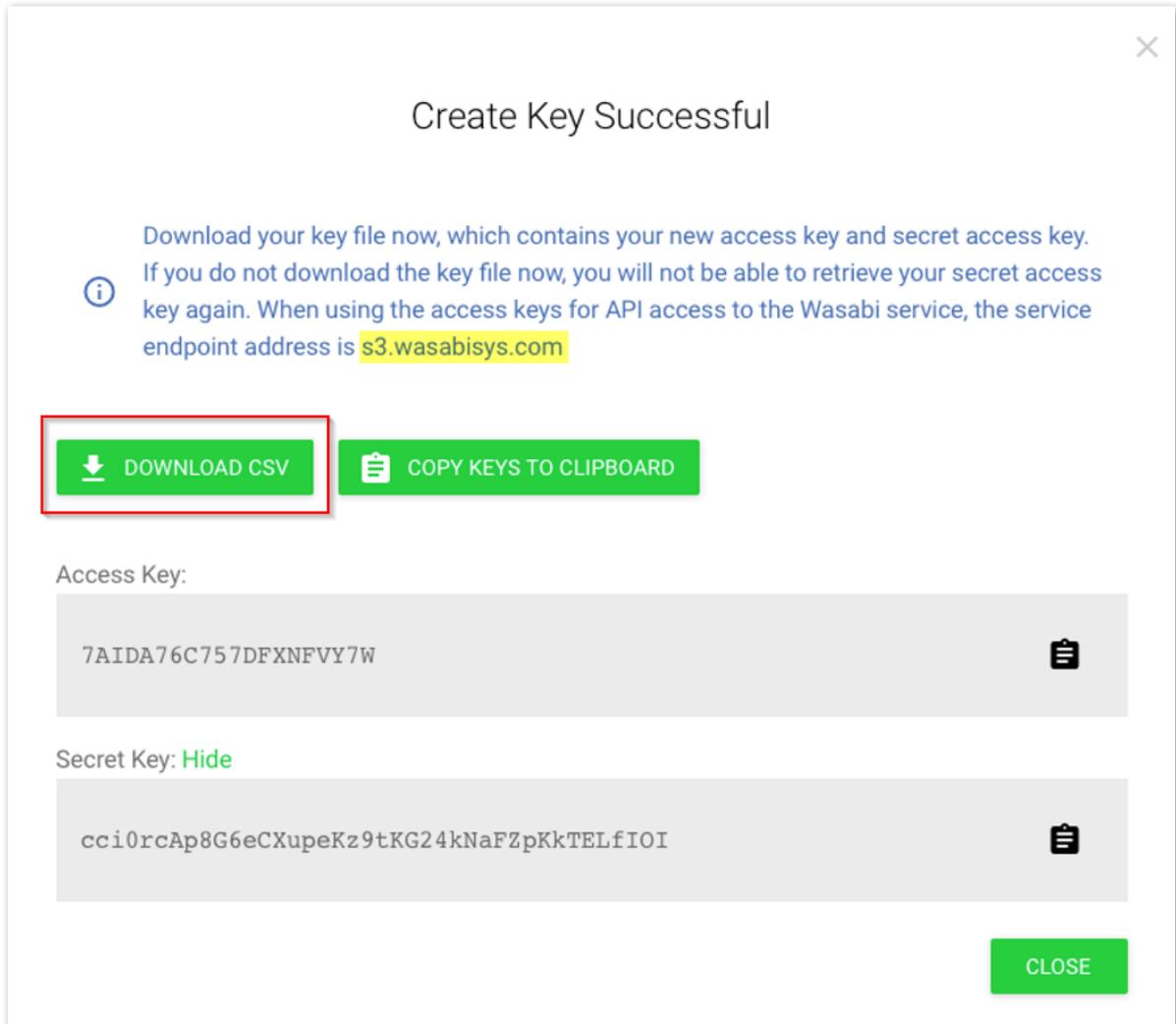
- a. Select the service **IAM** (Identity and Access Management) and click **Users** on the Wasabi console.



- b. Add a dedicated user, such as `caringo_backup`, to provide **Programmatic access** for Swarm.



- c. The Wasabi console generates an access key (an access key ID + secret access key), which must be recorded immediately.



- *The secret access key is not retrievable or viewable after this, so save it in a secure place.*

- d. Record the fully qualified user (such as "arn:aws:iam::123456789012:user/caringo_backup") for use later, in policies.
- e. Record these values for configuring the S3 Backup feed in Swarm:

- **Access Key ID**
- **Secret Access Key**

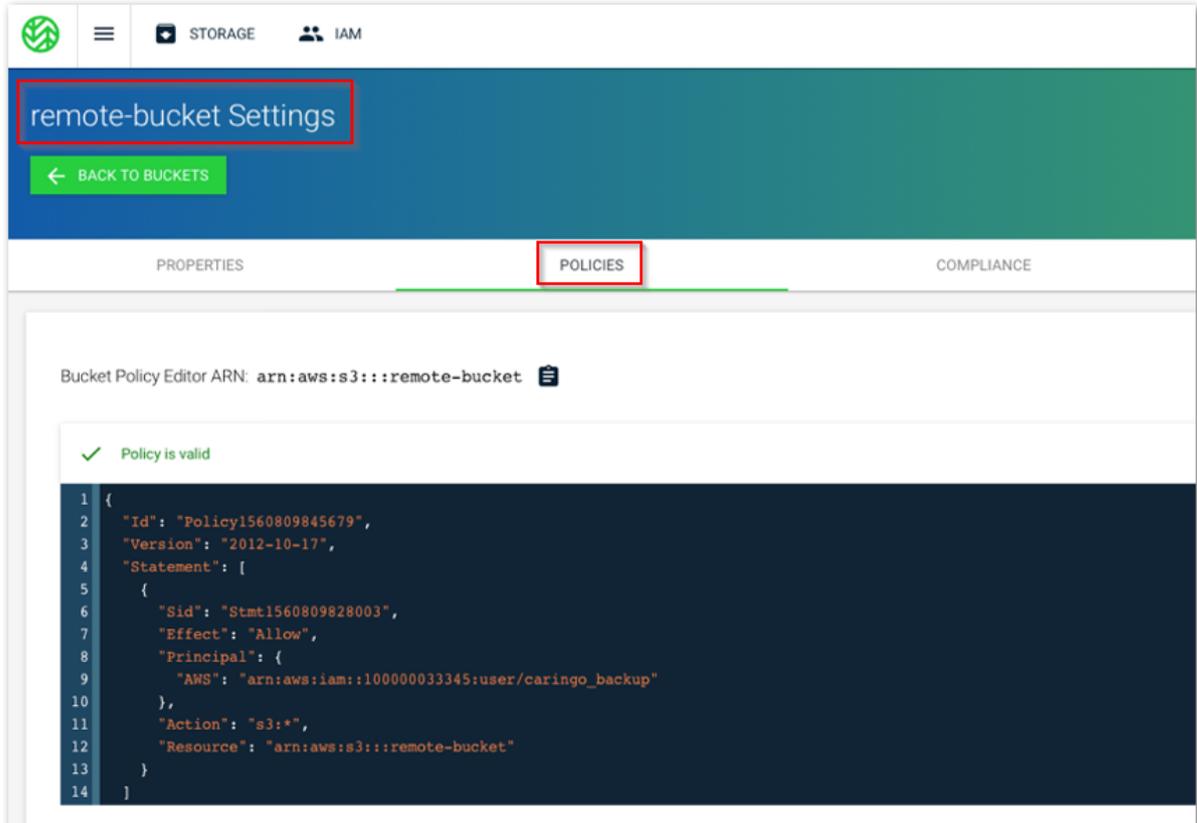
4. **Policies** – Create policies on *both* the user and the bucket so the programmatic user has exclusive rights to the S3 bucket. Use the policy generators provided or enter edited versions of the examples below.

- a. Create an **IAM policy** for this user, allowing it all S3 actions on the backup bucket, which needs to be specified as a fully qualified Resource (recorded above), starting with `arn:aws:s3:::`

IAM policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::example.cluster1.backup"
    }
  ]
}
```

- b. Create a matching **bucket policy** to grant access to the dedicated backup user, which needs to be specified as a fully qualified Principal, which is the User ARN (recorded above) starting with `arn:aws:iam::`. Using the Policy Generator, allow *all* S3 actions for a bucket, using the full ARN name:



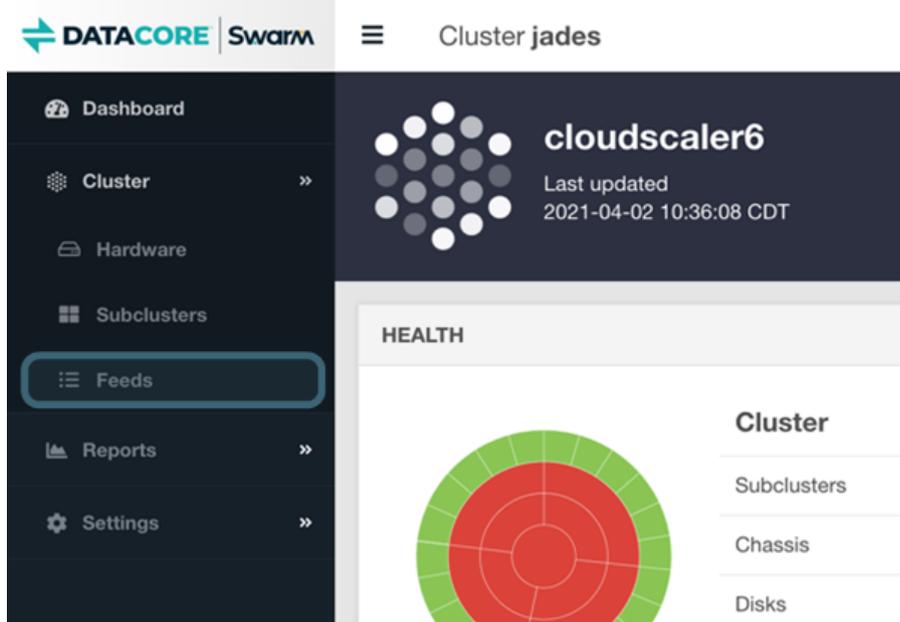
Bucket policy

```
{
  "Id": "Policy1560809845679",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1560809828003",
      "Action": "s3:*",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::example.cluster1.backup",
      "Principal": {
        "AWS": [
          "arn:aws:iam::123456789012:user/caringo_backup"
        ]
      }
    }
  ]
}
```

Configuring the S3 Backup Feed

Create a new S3 backup feed with Wasabi as the target on the Swarm side.

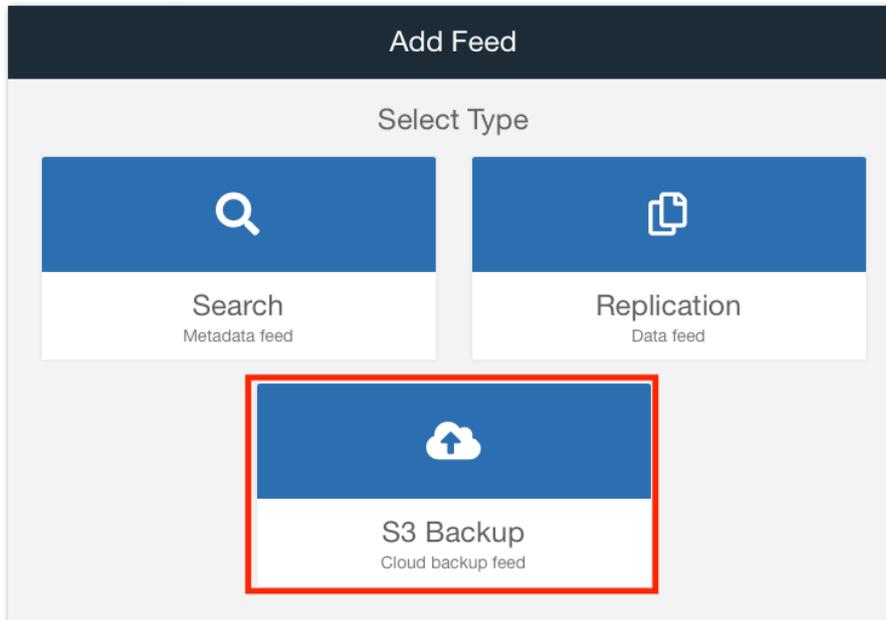
1. Open the **Feeds** page in the Swarm UI.



2. Click **+ Add** at the top right.



3. Choose the feed type **S3 Backup**.



4. Provide the following values. For details on these fields, see [S3 Backup Feeds](#).

- **Name** - For description, such as "Replication to Wasabi"
- **Endpoint** - Include the complete endpoint, without the bucket: s3.us-west-1.wasabisys.com
- **Region** - This example uses the **us-west-1** storage region. Use the [appropriate Wasabi service URL](#) for any other region.
- **Bucket** - Enter the newly created bucket dedicated to backing up the Swarm cluster.
- **Credentials** - Paste in both the Access key name and Secret Key
- **Use SSL** - Yes

S3 Backup Feed Revert Save

Name:

Scope: Entire source cluster (global) Only objects in select domain(s) Backup the entire Swarm cluster or specific storage domains

Propagate deletes: Enabled

Target S3 Provider

Endpoint: Do not include the bucket in the endpoint

Region:

Bucket:

Credentials:

Use SSL: Yes No

Port:

Local Cluster Forward Proxy: None Use proxy

Threads:

[Replicate to any Wasabi region and bucket](#)

5. Verify the new S3 backup appears in the list of Swarm feeds:

Cluster jades

Feeds 3

Feed	Status	Events queue	Deletes queue	Rate	Scope	Target
Search – Primary navys	Active	3	0	5,160.0 /hour	Global	navys
S3 backup S3 backup whole cluster	Paused	33,973,010	5,589	0 /hour	Global	
S3 backup Replication to Wasabi	Active	32,542,662	1,878	0 /hour	Global	s3.us-west-1.wasabisys.com us-west-1/my-remote-bucket

S3 Backup Restore Tool

The S3 Backup Restore Tool is the standalone utility for performing DR from the S3 backup bucket, either to the original cluster or to an empty cluster that is meant to replace the original. See [S3 Backup Feeds](#).

Once the data is backed up in S3, the restore tool allows both examining a backup and control how, what, and where it is restored:

- List all domains and buckets, or the buckets within a domain, with the logical space used for each.
- List all objects within a bucket or unnamed objects in a domain, optionally with sizes and paging.
- Restore either the complete cluster contents or else a list of domains, buckets, or individual objects to restore.
- Rerun the restore, should any part of it fail to complete.
- Partition the restoration tasks across multiple instances of the command line tool, to run them in parallel.
- [Installing the Restore Tool](#)
 - [Required](#)
 - [Preparation \(one-time\)](#)
 - [Installation](#)
 - [Restore Tool Settings](#)
 - [sample-swarmrestore.cfg](#)
 - [Additional Restore Configuration](#)
- [Using the Restore Tool](#)
 - [Full cluster restore](#)
 - [Specifying objects](#)
 - [ls subcommand](#)
 - [Note](#)
 - [restore subcommand](#)
 - [Note](#)
 - [Audit headers](#)

Installing the Restore Tool

The S3 Backup Restore tool has a separate install package included in the Swarm download bundle. Install it on one or more (for parallel restores) systems where the restore processes run.



Required

The S3 Backup Restore Tool must be installed on a system that is running RHEL/CentOS 7.

Preparation (one-time)

The **swarmrestore** package is delivered as a Python pip3 source distribution. Each machine needs to be prepared to be able to install this and future versions of swarmrestore.

1. As root, run the following command:

```
yum install python3
```

2. Verify version 3.6 is installed:

```
python3 --version
```

3. Upgrade pip

```
pip3 install --upgrade pip
```

Installation

Uninstall Python 2 generation of the tool (`caringo-swarmrestore-1.0.x.tar.gz`) if installed:

```
pip uninstall caringo-swarmrestore
```

Rerun this installation when a new version of `swarmrestore` is obtained:

1. Copy the latest version of the `swarmrestore` package to the server.
2. Run the following as root:

```
pip3 install caringo-swarmrestore-<version>.tar.gz
```

3. `swarmrestore` is likely in `/usr/local/bin` and is already in the path.
4. Repeat for any additional servers if planning to perform partitioning for parallel restores.

Restore Tool Settings

The tool uses a configuration file, `.swarmrestore.cfg`. Because the file contains sensitive passwords, the tool warns if the configuration file is not access-protected (`chmod mode 600` or `400`).

The configuration file follows the format of Swarm Storage settings files, using sections listing `name = value` pairs. These setting names map to the S3 Backup feed definition, where the values have the same meaning.

1. Locate the sample configuration file where it is installed:

```
/usr/local/sample-.swarmrestore.cfg
```

2. Copy the file into the home directory and rename it, and open it for editing:

```
cp /usr/local/sample-.swarmrestore.cfg ~/.swarmrestore.cfg
vi ~/.swarmrestore.cfg # Edit config settings
```

sample-.swarmrestore.cfg

```
# This is a sample configuration file for the swarmrestore utility.
# Save this file as ~/.swarmrestore.cfg and chmod 600 ~/.swarmrestore.cfg to keep passwords private.

# S3 host must be a fully qualified host name. The virtual host access style is supported if
# the host's first component is the bucket name.
# See https://docs.aws.amazon.com/general/latest/gr/rande.html#s3_region for Amazon S3 endpoints.
[s3]
host=s3.amazonaws.com
port=443
accessKeyID=<youraccesskeyid>
secretAccessKey=<yoursecretaccesskey>
bucketName=<yourbucketname>
region=us-east-1
# The option below uses HTTPS for access. For HTTP, set sslOption=none and adjust port.
sslOption=trusted

# The 4 options below are for swarmrestore initiating archival restore of content, such as GLACIER
performArchiveRetrieval=false
retrievalTier=Standard
accountID=<ninedigitaccountid>
activeLifetimeDays=7

# Use these only if you need a forward proxy to reach the S3 service.
[forwardProxy]
host=
port=80
username=
password=

# The log file can be /dev/null, but logs are useful for diagnosing problems.
[log]
filename=swarmrestore.log
level=30

# The Swarm cluster must either be directly accessible or accessible via
# a proxy. The password below is the administrative password for the cluster.
[swarm]
host=<space separated list of swarm host IPs or gateway host>
password=ourpwdofchoicehere
cluster=<yourclustername>
```

Section	Settings
[s3]	<ul style="list-style-type: none"> • host – The hostname of the S3 service. • port – The port to use for the S3 service. Use 443 or else 80, if SSL (sslOption) is disabled. • accessKeyID – The S3 access key ID. • secretAccessKey – The S3 secret access key. • bucketName – The name of the destination bucket in S3. • sslOption – The S3 connection constraint, with one of two values: <ul style="list-style-type: none"> • "trusted" (the default) specifies use of SSL and requires a trusted server certificate from the destination server. • "none" disables use of SSL. Use for testing and troubleshooting, and change the port to 80.

<p>[s3]</p> <p><i>archival</i></p>	<p>Set these additional parameters if using an S3 bucket with an archival storage class (Glacier, Glacier Deep Archive):</p> <ul style="list-style-type: none"> • performArchiveRetrieval – Whether restoration from archival storage is needed. Performing a restore does not incur any expenses for the bucket owner if false (<i>default</i>), . • retrievalTier – Which S3 Glacier retrieval tier to use for restoration: 'Standard' (<i>default</i>), 'Expedited', or 'Bulk'. Each tier has its own cost and expected restoration time; see Amazon S3 Storage Classes. • accountID – Specifies the 9-digit AWS account ID of the bucket owner, granting the tool permission to incur archive restoration expenses at the tier requested. This setting appears in the x-amz-expected-bucket-owner header on the restore object request. • activeLifetimeDays – How many days an object restored from archive should remain active before expiring (returning to archival storage). The default is 7 (1 week).
<p>[forwardProxy]</p>	<p>This section is for use with an optional forward proxy:</p> <ul style="list-style-type: none"> • host – The forward proxy hostname or IP address. • port – The forward proxy host to use. • username – (optional) The user name. • password – (optional) The password.
<p>[log]</p>	<p>The same log settings as the Swarm cluster may be used; identify the logs by looking for those with the component "RESTORE" if done so.</p> <ul style="list-style-type: none"> • host – The log host. Leave blank to disable logging. • port – (optional) The log port. Defaults to 514. • file – (optional) The log filename. Accepts the value of "stdout" for logging to the console screen. Defaults to <code>/dev/null</code>. • level – The log level. Defaults to 30 (Warning). Levels are the same used by Swarm: 20 (Info), 15 (Audit), 10 (Debug).
<p>[swarm]</p>	<ul style="list-style-type: none"> • host – A list of host names or IP addresses of Swarm nodes or Gateway nodes. • port – (optional) The SCSP port. Defaults to 80. • user – The cluster administrator user name, usually "admin". • password – The cluster administrator password.

Additional Restore Configuration

- **Gateway** – Add the IP of the machine where the Restore tool runs to the [Gateway configuration setting](#) `scsp.allowSwarmAdminIP` if communicating with a Swarm cluster via Gateway.

Using the Restore Tool

i Full cluster restore

Before undertaking a restore of a large cluster, contact DataCore Support. They help balance the speed of the restore with bandwidth constraints by examining the space used by the S3 backup bucket, estimating the bandwidth needed, and recommending best use of the `-p` command line option (for multiple simultaneously running restore commands on different hosts). They also advise on whether a forward proxy is needed, to reduce bandwidth usage.

The AWS bucket may be pulled out of cold storage before the full cluster restore by changing the storage class to Standard if using an AWS Glacier storage class.

The restoration tool runs using batch-style operation with commands given on the command line. The tool logs the actions to the log file or server in the log configuration section. The restoration tool uses the following command format:

```
swarmrestore [<tool option>...] <command> [<command option> ...] [<objectspec> ...]
```

i Specifying objects

`<objectspec>`, or *object specification*, refers to how the path to the Swarm object to be targeted is referenced. It may be a domain name, a bucket name, a named object, an unnamed UUID, or an historical version of an object.

Options:

- `--help` – Displays a summary of the current configuration.

```
>> swarmrestore --help
usage: swarmrestore [-h] [-v] {ls,restore} ...
```

Explore or restore objects stored in an S3 backup of a Swarm cluster.

positional arguments:

```
{ls,restore}
  ls                list the contents of the S3 bucket, optionally recursively or
                    using a long format
  restore           restore the contents of the S3 bucket, optionally recursively
                    or including prior versions
```

optional arguments:

```
-h, --help        show this help message and exit
-v, --version     show program's version number and exit
```

Uses `~/.swarmrestore.cfg` for configuration.

- `--version` – Reports the version of the tool.
- `ls --help` – Displays help on the `ls` command, for listing and enumerating.

```
>> swarmrestore ls --help
usage: swarmrestore [-h] [-v] {ls,restore} ...
```

Explore or restore objects stored in an S3 backup of a Swarm cluster.

positional arguments:

```
{ls,restore}
  ls                list the contents of the S3 bucket, optionally recursively or
                    using a long format
  restore           restore the contents of the S3 bucket, optionally recursively
                    or including prior versions
```

optional arguments:

```
-h, --help          show this help message and exit
-v, --version       show program's version number and exit
```

Uses `~/.swarmrestore.cfg` for configuration.

- `restore --help` – Displays help on the `restore` command, for selective restore and disaster recovery.

```
>> swarmrestore restore --help
usage: swarmrestore restore [-h] [-R] [-v] [-n] [-p count/total] [-f FILE]
                             [objectspec [objectspec ...]]
```

positional arguments:

```
objectspec          any number of object specifications to restore
```

optional arguments:

```
-h, --help          show this help message and exit
-R, --recursive     recursively traverse the objectspecs
-v, --versions      also restore prior versions
-n, --noop          perform checking but do not actually restore
-p count/total, --partition count/total
                    partition the work <count> from among <total>
-f FILE, --file FILE use the specified file for objectspecs, one per line
```

ls subcommand

Enumeration and selection are handled by the `ls` command, which is modeled after the Linux command `ls` and whose results are captured with standard Linux `stdout`. Use the command to visualize what domains and buckets are backed up in S3 and are available to be restored. The output is sorted by name and interactively paginated to help manage large result sets by default.

The `ls` subcommand has this format:

```
ls [<command option> ...] [<objectspec> ...]
```

Command options, which can be combined (for example, `-Rvl`):

- `-R` or `--recursive` – Recursively lists the given domain or bucket, or else the entire cluster. Without this option, the command lists the top-level contents of the object.
- `-v` or `--versions` – List previous versions of versioned objects. Versions are not listed by default.
- `-l` or `--long` – Lists details for each item returned in the output:
 - Creation date
 - Content length of the body
 - ETag
 - Archive status:
 - AN – Archived; not available for restoration
 - AR – Archived with an archive restore in progress; not available for restoration

- AA – Archived with a copy available for restoration
- OK – Not archived and fully available
- Objectspec
- Alias UUID, if the object is a domain or bucket
- `<objectspec>` – If none, the command runs across the entire contents of the S3 backup. If present, filters the command to a specific domain or bucket (*context object*) in Swarm. Use this format:

Cluster	
Domain	mydomain/
Bucket	mydomain/mybucket/
Named object	mydomain/mybucket/myobject/name/with/slashes.jpg
Named version	mydomain/mybucket/myobject/name/with/slashes.jpg//645f3912802bb4c31311afc46de2cfc3
Unnamed object	mydomain/06ea262a860af23504261f50c09a6b29 (<i>no domain if untenanted</i>)
Unnamed version	mydomain/06ea262a860af23504261f50c09a6b29//137a88d550041ecda9b8ec4bc36ebea2

Note

Use the double-slash format (//) before including a specific version ID for an object. Newlines separate objects.

When running the command without any options, it returns the list of domains that are included in this S3 bucket for the Swarm cluster:

```
>>> swarmrestore ls
domain1/
domain2/
www.testdomain.com/
```

Run a command like this if wanting a complete accounting of every object backed up for a specific domain, redirecting to an output file:

```
>>> swarmrestore ls -Rvl mydomain/ > mydomaincontents
```

restore subcommand

Object restoration and verification is handled by the `restore` subcommand, which has the following format:

```
restore [<command option> ...] [<objectspec> ...]
```

`<objectspec>` – If none, applies the command to the *entire cluster backup*. If present, filters the command to a specific domain, bucket, object, or object version.

To target a command to a specific context (domain/bucket) or content object in Swarm, format the type of object as follows:

Cluster	
Domain	mydomain/

Bucket	mydomain/mybucket/
Named object	mydomain/mybucket/myobject/name/with/slashes.jpg
Named version	mydomain/mybucket/myobject/name/with/slashes.jpg //645f3912802bb4c31311afc46de2cfc3
Unnamed object	mydomain/06ea262a860af23504261f50c09a6b29 (no domain if untenanted)
Unnamed version	mydomain/06ea262a860af23504261f50c09a6b29//137a88d550041ecda9b8ec4bc36e2



Note

Use the double-slash format (//) before including a specific version ID for an object. Newlines separate objects.

Any number of command options can be used, and the short forms may be combined with a single dash (-Rv). The <objectspecs>, -R, and -v options iterate over objects the same way as the `ls` command.

Options:

- `-R` or `--recursive` – Recursively restore domains, buckets, or the entire cluster with an empty object spec. See above for what is iterated over when -R is not used.
- `-v` or `--versions` – Include previous versions of versioned objects. They are not included by default.
- `-f <file>` or `--file <file>` – Use objectspecs from a file instead of the command line.
- `-p <count>/<total>` or `--partition <count>/<total>` – Partition work for a large restore job (but every instance restores buckets and domains before objects).
 - Example: To run 4 instances in parallel, configure each option to be one of the series: `-p 1/4`, `-p 2/4`, `-p 3/4`, `-p 4/4`
- `-n` or `---noop` – Perform the checking of a restore, but do not restore any objects.
 - Does not change the cluster state. The option can be used before and after a restore, as both a pre-check and a verification.
- `<objectspecs>` – Any number; newlines separate objects. If none, the top level of the cluster's backup contents is the scope.
 - Using no object specification with the command options `-Rv` causes Swarm to restore *all backed up objects in the entire cluster*, including any historical versions of versioned objects.

What is restored: Restore copies an object from S3 to the cluster if the cluster object is missing or else older than the S3 object. Note: context objects restore before the content they contain: restore first restores any domains or buckets needed before restoring objects within them.

Output of Restore – At the end of the restoration, the tool reports the number of objects restored and the number of objects skipped, for being either identical to or newer than the backed up copy. The command output lists each object spec with its status:

- `current` – The object was not restored because the target cluster already has the same version of the object.
- `older` – The object was not restored because it is older than the one in the target cluster.
- `obsolete` – The object was not restored because the cluster does not allow the object to be written. Usually it means the object is deleted.
- `needed` – The object needs restoration, but the -n option was used.
- `restored` – The object was successfully restored.
- `nocontext` – The object cannot be restored because its parent domain or bucket cannot be restored.
- `failure` – The object cannot be restored. Consult the logs for details.

- `archived` – The object is archived and the restore tool is not configured for archive restoration. This is a failure condition.
- `initiated` – The object is archived and the tool has issued an object restoration request. See the [Amazon S3 API RestoreObject Request Syntax](#). This is also a failure condition, but the object is counted in the archive retrieval initiated stats. It is these operations that incur expense to the bucket owner by the restore tool.
- `ongoing` – The object is in archive and a restoration request has already been initiated. Restoration from archive is in progress. This is also a failure condition.

Rate of Restore – Restoration may take a long time run, especially if recursion (`-R`) is used on domains or buckets. To boost the rate of restore, install the S3 Backup Restore tool on multiple servers and then run the restore command with partitioning parameters (`-P`) across all instances of the tool, which allows restoring faster in parallel, with minimal overlap.

Headers for Audit – When the S3 Backup feed writes an object to the S3 bucket, it adds to the S3 copy a header (**Castor-System-Tiered**) that captures when and from where the object was tiered. When the S3 Backup Restore tool writes the S3 object back to Swarm, it includes that S3 header and then adds another one of the same, to capture when and from where the object was restored. These paired headers (both named **Castor-System-Tiered**) provide the audit trail of the object's movement to and from S3. Swarm persists these headers but does not include them in Entity-MD5 or Header-MD5 calculations. The dates are of the same format as **Last-Modified** ([RFC 7232, section 2.2](#)). See [SCSP Headers](#).

Audit headers

```
Castor-System-Tiered: <date-of-backup> <cluster-name>/<cluster-settings-uuid>
Castor-System-Tiered: <date-of-restore> <S3-service-host>/<bucket-name>
```

S3 Backup Feeds to RStor

- [Backup](#)
- [Setting up the S3 Bucket](#)
 - [Note](#)
- [Configuring the S3 Backup Feed](#)

Backup

Objects in the S3 backup bucket are wholly dedicated to disaster recovery for Swarm and are not for general use by owners of the account where the bucket resides. Consider this feature a restricted form of S3, with constraints on the bucket's namespace that support Swarm's ability to backup and restore. For this reason, do not expect the namespace to be end-user friendly.

Swarm S3 backups to RStor targets are verified. To implement an S3 backup feed, first complete a one-time set up of the destination: set up an account with RStor and then create an S3 bucket dedicated to backing up *this cluster* .

Setting up the S3 Bucket

Note

Swarm has the RStor access granted as part of this configuration. Neither the S3 Backup feed nor the [S3 Backup Restore Tool](#) administers the S3 credentials or creates any S3 buckets in RStor. See [RStor Support](#) for assistance.

These instructions are for [RStor](#) cloud storage, but any Internet-based S3 service has similar functionality:

1. **Service** – If needed, sign up for RStor: rstor.io
2. **Bucket** – Create a bucket dedicated to backing up the Swarm cluster.
 - a. Sign in to the RStor console: rstorcloud.io.
 - b. Choose **Add new bucket**.
 - c. For **Bucket name**, enter a DNS compliant name for the new bucket. Buckets cannot be renamed, so choose wisely:
 - The name must be unique across all existing bucket names in RStor S3.
 - The name must be a valid DNS name, containing lowercase letters and numbers (and internal periods, hyphens, underscores), between 3 and 64 characters.
(See S3 documentation: [Rules for Bucket Naming](#).)
Tip: For easier identification, incorporate the name of the Swarm cluster this bucket is dedicated to backing up.
 - d. For **Region**, choose the one appropriate for the business needs.
 - e. *Best practice:* Do not enable versioning or any other optional features unless it is required for the organization.
 - f. Record these values for configuring the S3 Backup feed in Swarm:
 - **Bucket Name**
 - **Region**
 - g. Click **Submit**.

Add new bucket

Choose a name for new bucket, the replication policy and whether the bucket will be public or not.

Name of the bucket *
caringo-dr

Bucket name must be globally unique. Between 3 - 63 chars. Lowercase letters, digits and - allowed. Must start with a char. 10 / 63

Regions
Select the region(s) where you want the bucket to be created and replicated.

us-east-1 us-central-1 us-west-1

Access Mode
Private

Versioning
Keep previous versions of objects when they are overwritten or deleted.

Enabled Disabled

Object Locking
Store objects using a write-once-read-many (WORM) model to meet regulatory requirements or add an extra layer of protection against object changes and deletion.

Will enable versioning automatically.

Enabled Disabled

3. **S3 Key Pair**
—

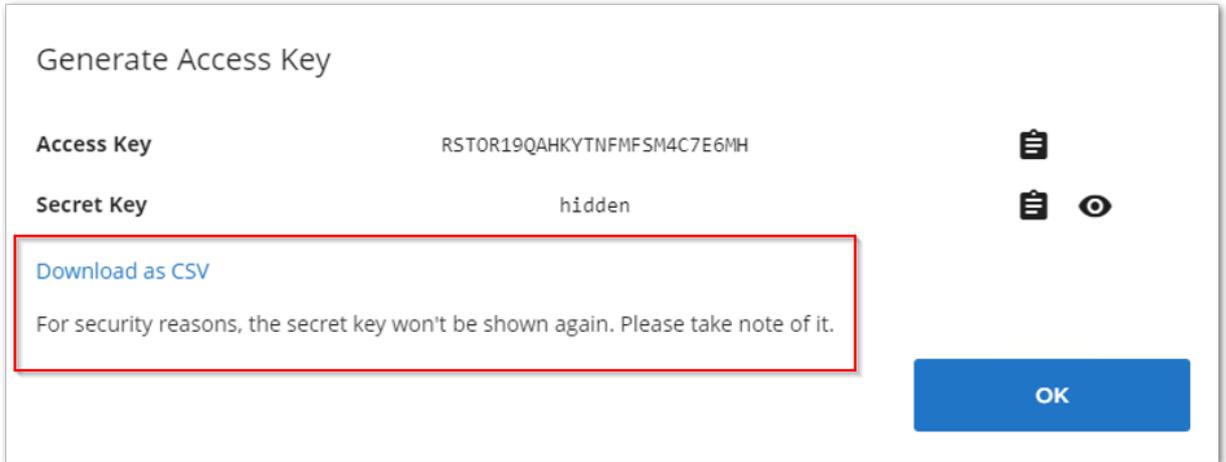
Create

- a. Click **My account** under the email address.
- b. Down below click **Generate Key**.

+ GENERATE KEY

ID	Generated On	Last Access	Actions
 RSTOR1P24RX40OCMM2IN3RZSMX	08/27/20 00:35:22 UTC	08/27/20 UTC	

- c. An S3 key pair is automatically created. Download and secure the access key:

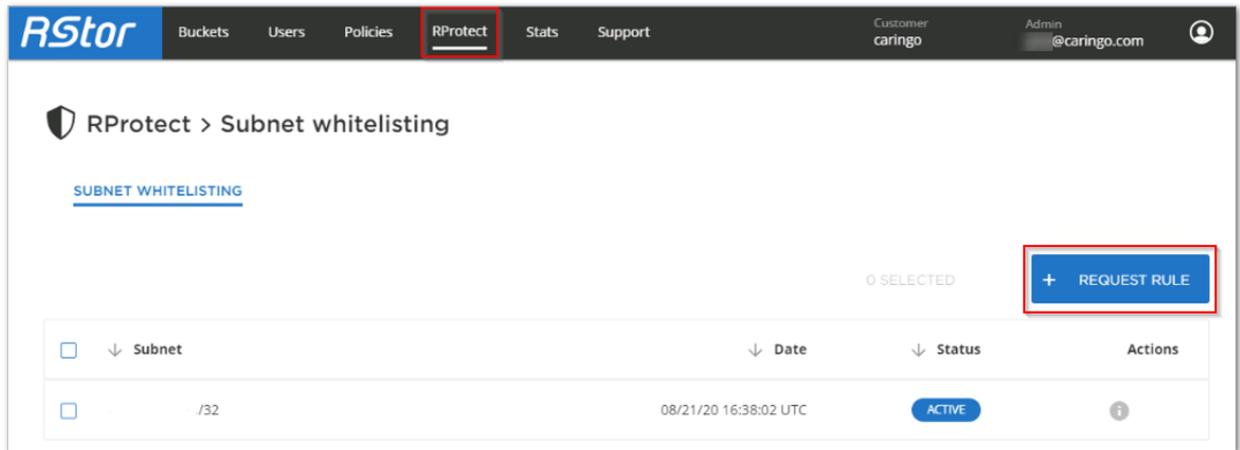


d.

- **Access Key ID**
- **Secret Access Key**

4. **RProtect** – Complete any subnet whitelisting needed.

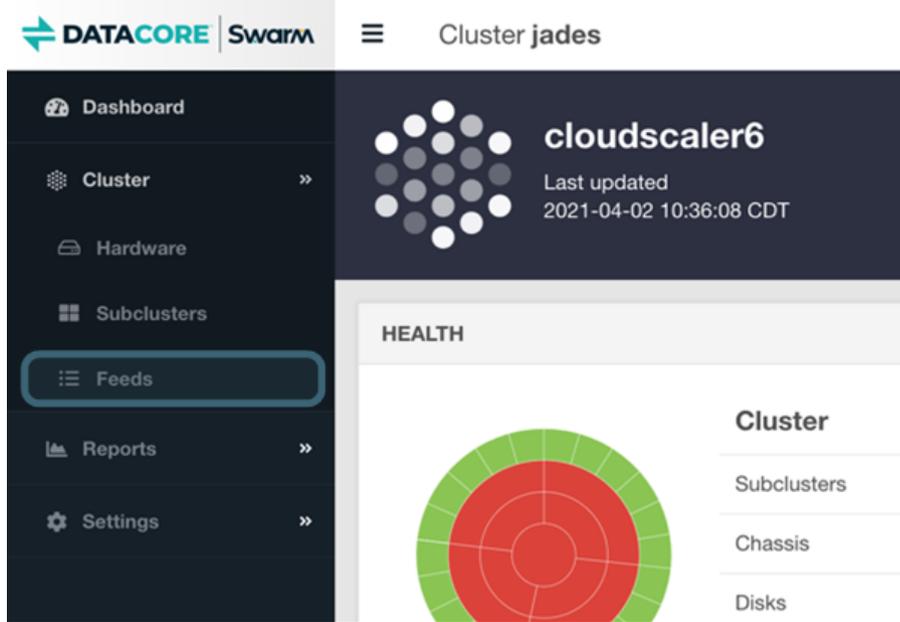
- Click on **RProtect** in the global menu bar.
- Verify the IP address of the Swarm cluster contacting the RStor S3 cloud service is allowed.
- Request a new rule if not the case.



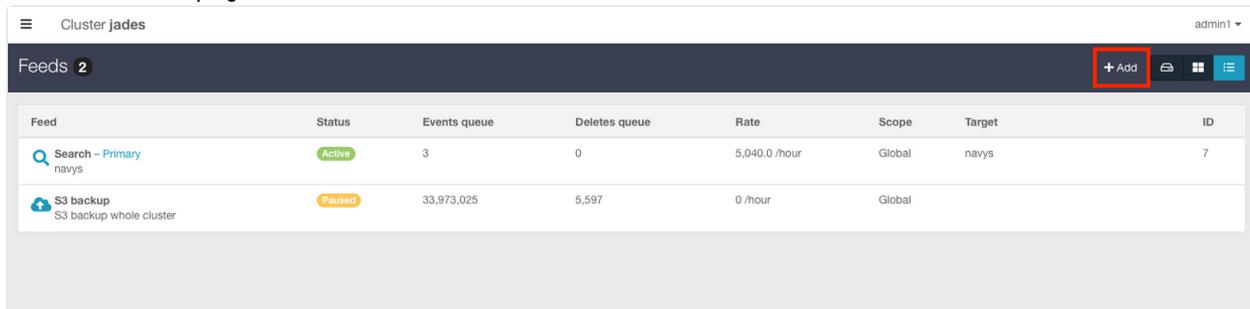
Configuring the S3 Backup Feed

Create a new S3 backup feed with RStor as the target on the Swarm side.

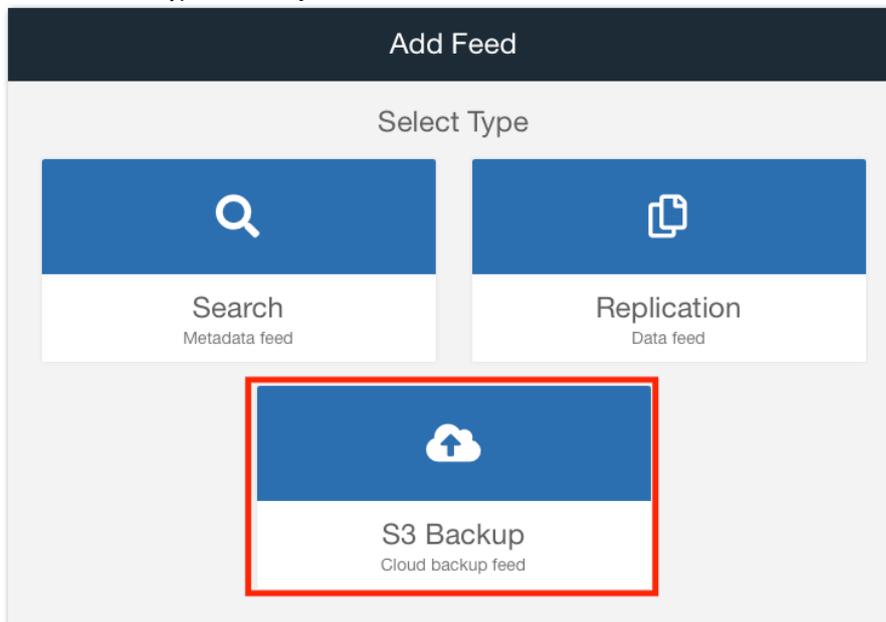
1. Open the **Cluster > Feeds** page in the Swarm UI.



2. Click **+ Add** at the top right.



3. Choose the feed type **S3 Backup**.



4. Provide the following values. See [S3 Backup Feeds](#) for details on these fields.

- **Name** - For description, such as "Replication to RStor"
- **Endpoint** - Include the complete endpoint, *without the bucket*: `s3.rstorcloud.io`
- **Region** - This example uses the **us-west-1** storage region. Check with RStor Support for any other region.
- **Bucket** - Enter the newly created bucket, which is dedicated to backing up the Swarm cluster.
- **Credentials** - Paste in both the Access key name and Secret Key
- **Use SSL** - Yes

S3 backup whole cluster

[Revert](#) [Save](#)

[See S3](#)

ID 12

Status Paused

Name

Scope Entire source cluster (global) Only objects in select domain(s)

Propagate deletes Enabled

Target S3 Provider

Endpoint

Region

Bucket

Credentials

Use SSL Yes No

for details on all field options.

[Backu](#)

5. Verify the new S3 backup appears in the list of Swarm feeds on the **Cluster > Feeds** dashboard.

Cluster jades				
Feeds 3				
Feed	Status	Events queue	Deletes queue	
Search - Primary navys	Active	3	0	
S3 backup S3 backup whole cluster	Paused	33,973,010	5,589	
S3 backup Rstor	Active	32,542,662	1,878	

See [Managing Feeds](#) for more details on feed administration in Swarm.

S3 Backup Feeds to Seagate Lyve

- [Backup](#)
- [Setting up the S3 Bucket](#)
 - [Note](#)
- [Configuring the S3 Backup Feed](#)

Backup

Objects in the S3 backup bucket are wholly dedicated to disaster recovery for Swarm and are not for general use by owners of the account where the bucket resides. Consider this feature a restricted form of S3, with constraints on the bucket's namespace that support Swarm's ability to backup and restore. For this reason, do not expect the namespace to be end-user friendly.

Swarm S3 backups to Seagate Lyve targets are verified. To implement an S3 backup feed, first complete a one-time set up of the destination: set up an account with Seagate Lyve and then create an S3 bucket dedicated to backing up *this cluster*.

Setting up the S3 Bucket

Note

Swarm has the Seagate Lyve access granted as part of this configuration. Neither the S3 Backup feed nor the [S3 Backup Restore Tool](#) administers the S3 credentials or creates any S3 buckets in Seagate Lyve.

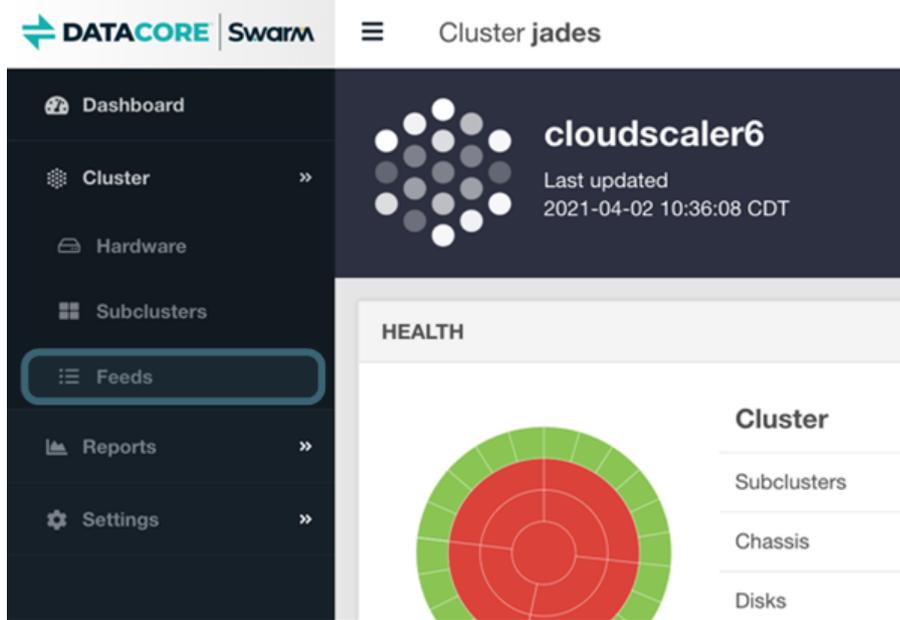
These instructions are for Seagate Lyve cloud storage, but any Internet-based S3 service has similar functionality:

1. **Service** – If needed, sign up for Seagate Lyve:
2. Record these values for configuring the S3 Backup feed in Swarm.
 - a. **Bucket Name**
 - b. **Endpoint**
 - c. **S3 Key Pair** – Create an S3 key pair dedicated to Swarm access.
 - **Access Key ID**
 - **Secret Access Key**

Configuring the S3 Backup Feed

Create a new S3 backup feed with Seagate Lyve as the target on the Swarm side.

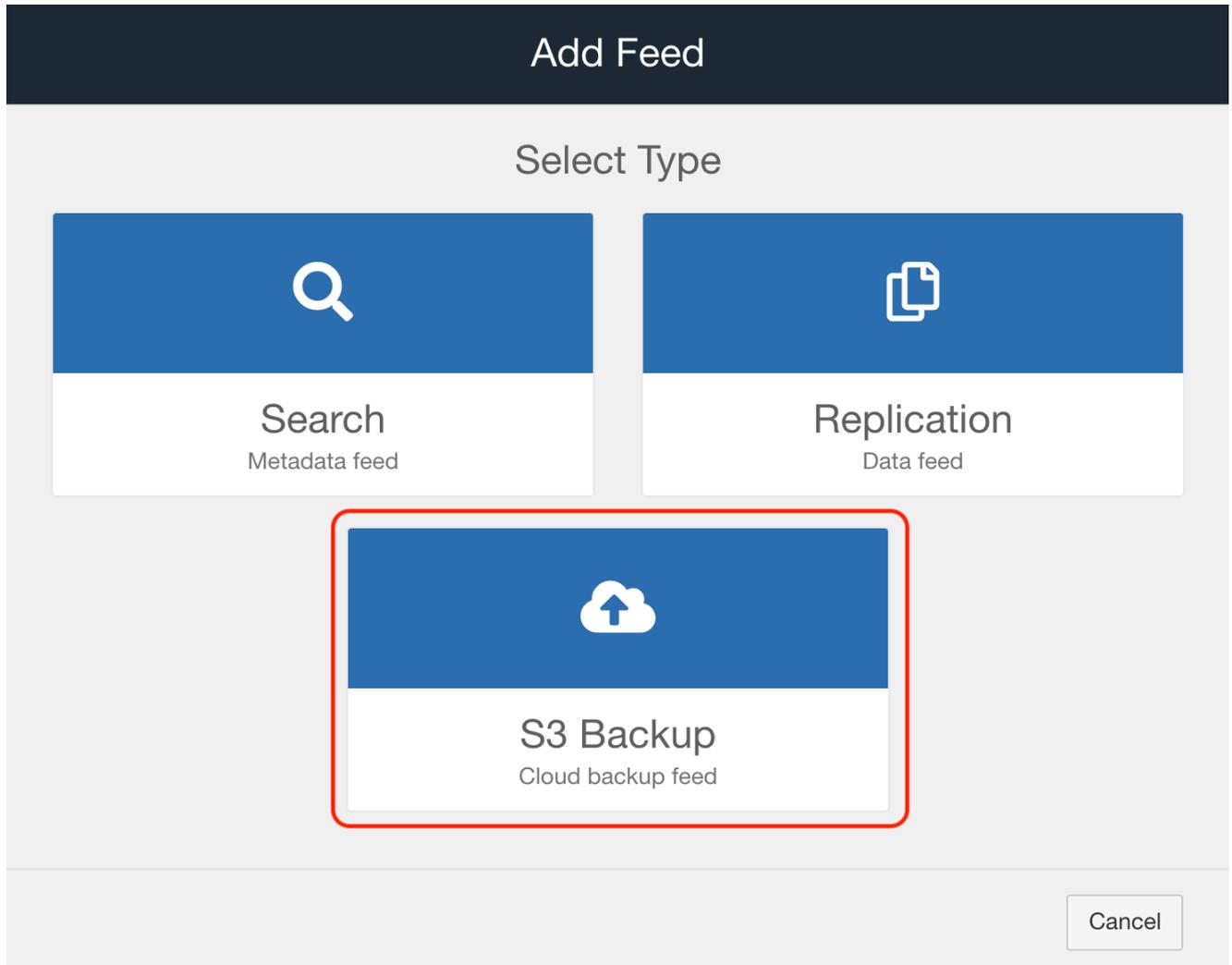
1. Open the **Cluster > Feeds** page in the Swarm Storage UI.



2. Click **+ Add** at the top right.

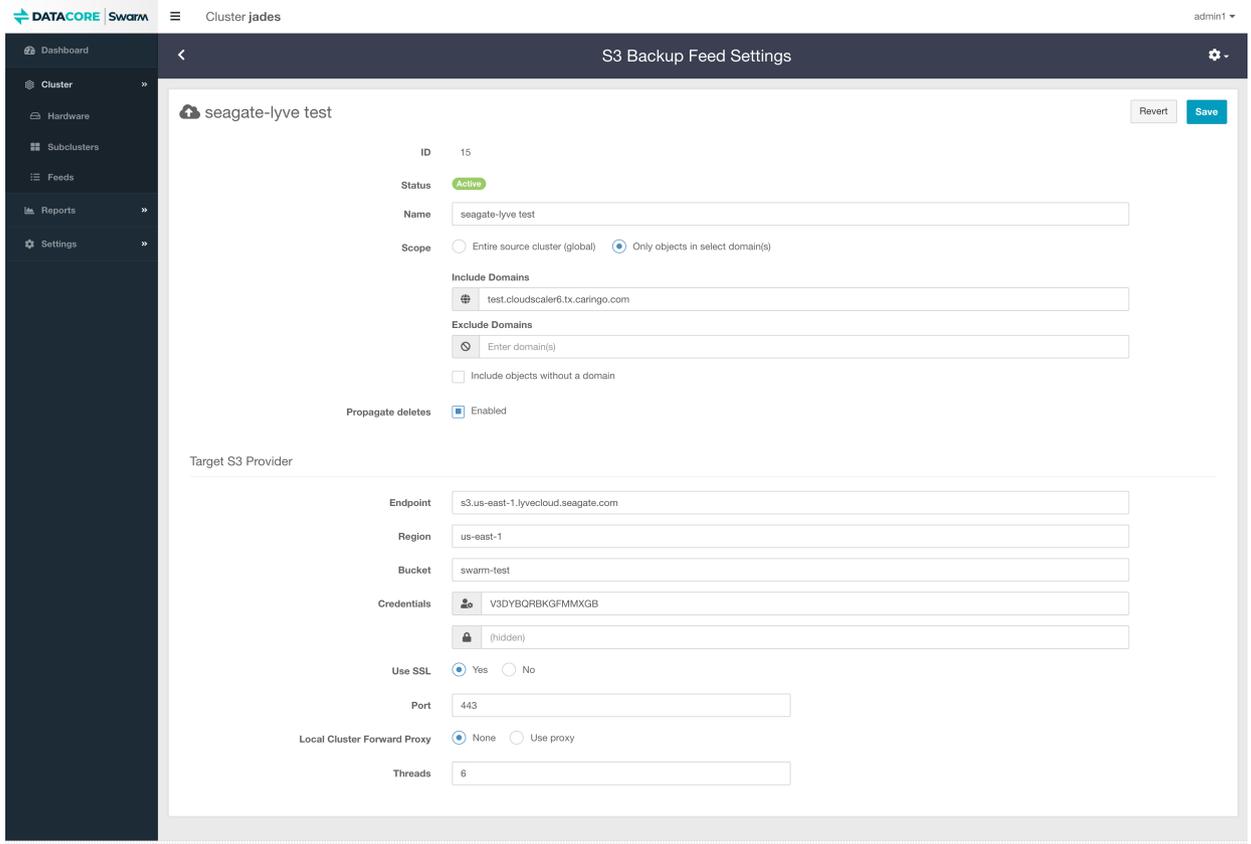


3. Choose the feed type **S3 Backup**.



4. Provide the following values. See [S3 Backup Feeds](#) for details on these fields.

- **Name** - For description, such as "Replication to Seagate Lyve"
- **Endpoint** - Include the complete endpoint, *without the bucket*: s3.us-east-1.lyvecloud.seagate.com
- **Region** - This example uses the **us-east-1** storage region. Check with Seagate Lyve Support for any other region.
- **Bucket** - Enter the newly created bucket, which is dedicated to backing up the Swarm cluster.
- **Credentials** - Paste in both the Access key name and Secret Key
- **Use SSL** - Yes



See [S3 Backup Feeds](#) for details on all field options.

5. Verify the new S3 backup appears in the list of Swarm feeds on the **Cluster > Feeds** dashboard.

 S3 backup seagate-lyve test	Active	51,332,809	295,457	0 /hour	Domain	s3.us-east-1.lyvecloud.seagate.com us-east-1/swarm-test	15
--	---	------------	---------	---------	--------	--	----

See [Managing Feeds](#) for more details on feed administration in Swarm.

S3 Backup Feeds to IBM Cloud Object Storage

- [Backup](#)
- [Setting up the S3 Bucket](#)
 - [Note](#)
- [Configuring the S3 Backup Feed](#)

i **Backup**

Objects in the S3 backup bucket are wholly dedicated to disaster recovery for Swarm and are not for general use by owners of the account where the bucket resides. Consider this feature a restricted form of S3, with constraints on the bucket's namespace that support Swarm's ability to backup and restore. For this reason, do not expect the namespace to be end-user friendly.

Swarm S3 backups to IBM cloud Object Storage targets are verified. To implement an S3 backup feed, first complete a one-time set up of the destination: set up an account with IBM and then create an S3 bucket dedicated to backing up *this cluster*.

Setting up the S3 Bucket

i **Note**

Swarm has the IBM Cloud Object Storage access granted as part of this configuration. Neither the S3 Backup feed nor the [S3 Backup Restore Tool](#) administers the S3 credentials or creates any S3 buckets in IBM. See IBM Support for assistance.

These instructions are for IBM Cloud Object Storage, but any Internet-based S3 service has similar functionality:

1. **Account** – Create an account at cloud.ibm.com or use the [IBM Cloud CLI](#) to log in and create resources if needed.
2. **Resources, Service Instances, and Buckets** – See [Working with resources and resource groups \(ibmcloud resource\)](#) and [Getting started with IBM Cloud Object Storage](#) for details on creating resources, service instances and buckets using either the IBM Cloud UI or IBM Cloud CLI.
3. **S3 Key Pair** – Create an S3 key pair dedicated to Swarm access. See IBM's guide on [Using HMAC credentials](#) for details.

Configuring the S3 Backup Feed

Create a new S3 backup feed with IBM Cloud Object Storage as the target on the Swarm side.

1. Open the **Cluster > Feeds** page in the Swarm UI.
2. Click **+ Add** at the top right.
3. Choose the feed type **S3 Backup**.
4. Provide the following values. See [S3 Backup Feeds](#) for details on these fields.
 - **Name** - For description, such as "Replication to IBM Cloud Object Storage"
 - **Endpoint** - Include the complete endpoint, *without the bucket*: `s3.us.cloud-object-storage.appdomain.cloud`
 - **Region** - This example uses the **us-geo** storage region. Check with IBM Support for any other region or see [IBM's list of endpoints](#).
 - **Bucket** - Enter the newly created bucket, which is dedicated to backing up the Swarm cluster.

- **Credentials** - Paste in both the Access key name and Secret Key
- **Use SSL** - Yes

See [S3 Backup Feeds](#) for details on all field options.

5. Verify the new S3 backup appears in the list of Swarm feeds on the **Cluster > Feeds** dashboard.
See [Managing Feeds](#) for more details on feed administration in Swarm.

Replication Feeds

- [Types of Replication](#)
- [Adding a Replication Feed](#)
- [Using Feed Actions](#)
- [Troubleshooting Feeds](#)

Types of Replication

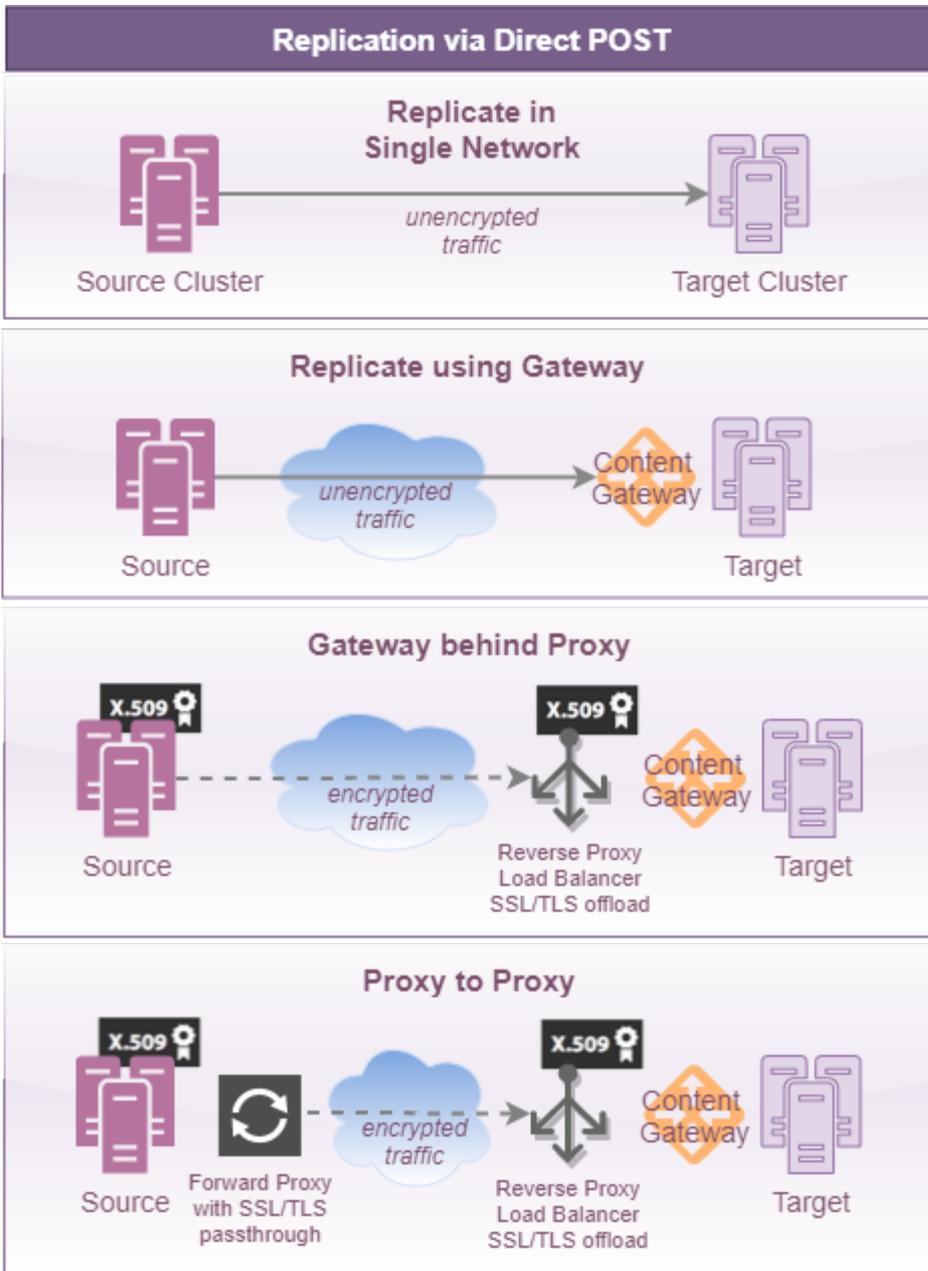
What type of replication method chosen and how to configure it depends on whether a legacy Swarm implementation exists and on the needs for securing replication traffic over untrusted networks. (v10.0)

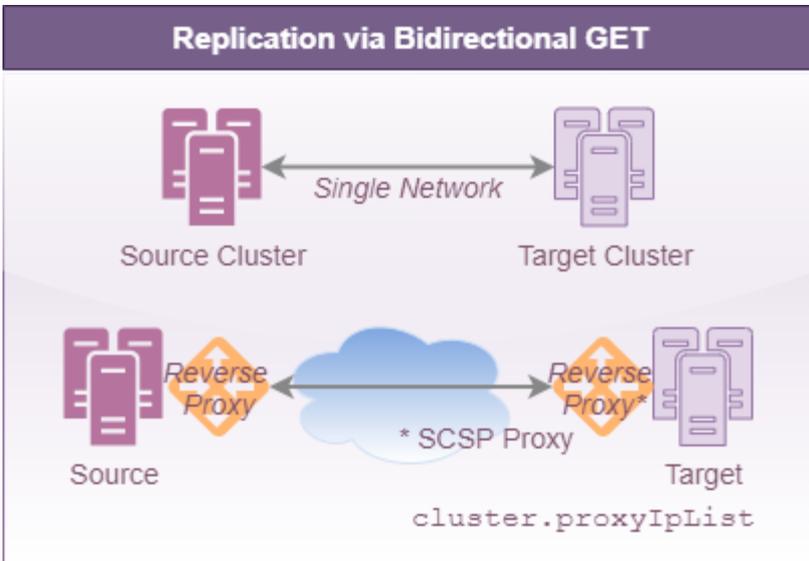
Secure Replication – Swarm Storage supports remote replication over a WAN, so replication feeds can operate through Content Gateway. When a replication feed is defined, specify which replication mode to use: either the legacy bidirectional GET method of replication (which may be needed for specific application compatibility or network requirements) or the recommended direct POST method, which offers better performance and flow management. With Swarm Storage 10.0 and later, TLS/SSL security can be implemented as fits the implementation:

- Upload a trusted certificate to Swarm
- Replicate to an SSL offloader that services the target cluster
- Replicate from a forward proxy on the source cluster.

See [Replicating Feeds over Untrusted Networks](#) and [Adding a Trusted Certificate to Swarm](#).

Replication Methods – Below are two replication methods available, along with the configuration variants of each that are supported. For best performance, choose direct POST replication, which can go through Gateway. GET replication is the legacy method, which may be needed for application compatibility or networking requirements.



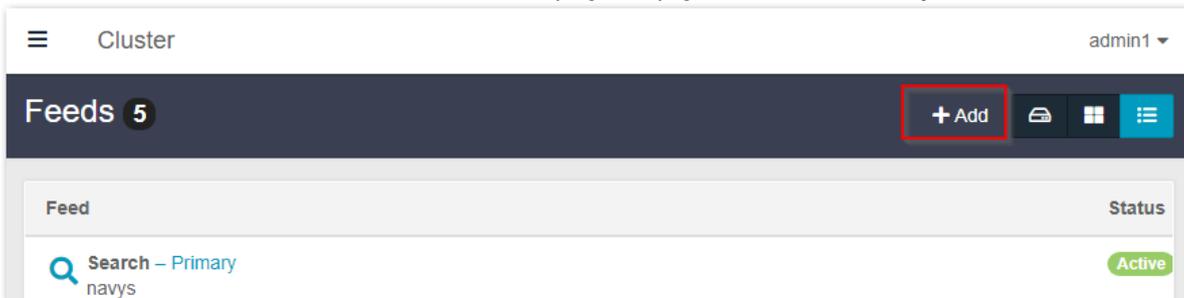


Note

Using the legacy Bidirectional GET for remote replication requires populating the [Storage configuration setting](#) `cluster.proxyIpList` for any cluster using a reverse proxy. The setting is a comma-separated list of reverse proxy IP addresses or names, including ports in `name:port` format. If using Direct POST replication, this setting can be populated or left blank, as it has no effect.

Adding a Replication Feed

To add a feed in the cluster, click the **+Add** button at the top right the page and then select the **Replication** button.



When a replication feed is defined, set the scope and select which type (*Replication Mode*) is in force and with what speed (number of concurrent *Threads*), if using direct POST:

Replication Feed
Revert Save

Name

Scope Entire source cluster (global) Only objects in select domain(s)

Propagate deletes Enabled

Disabling is deprecated. Use Object Versioning on the target cluster to be able to recover deleted objects.

Target Remote Cluster

Cluster Name

Proxy or Host(s)

Port

Replication Mode Replicate via bidirectional GET (if needed)
 Replicate via direct POST (recommended; supports Gateway)

Threads

SSL Server
 None Require trusted SSL Allow untrusted SSL (not recommended)

Remote Admin Name/Password Inherit from source cluster

The following table describes the data entry fields in the dialog box.

ID (existing feeds)	Read-only; system-assigned identifier
Status (existing feeds)	Read-only; the current feed processing state.
Name	The name attached to a feed.

<p>Scope</p>	<p>The feed filters that selected for the replication feed. The object is replicated to the domain(s) indicated in this field.</p> <div data-bbox="272 264 1485 562" style="border: 1px solid #ccc; padding: 10px; margin-bottom: 10px;"> <p>i Gateway adminDomain</p> <p>Do not <i>create</i> the same domain in two clusters: always create it in the source cluster and then <i>replicate</i> it to the target. A Gateway must use an independent adminDomain, at least temporarily, if the Gateway is in front of the target cluster. (CLOUD-2785)</p> <p>Verify the source cluster's adminDomain is included in the list of replicated domains if choosing to replicate specific domains.</p> </div> <ul style="list-style-type: none"> • Entire source cluster (global) – To replicate <i>all</i> objects in the source cluster, leave the default selection of Entire source cluster (global) • Only objects in select domain(s) – To replicate the objects in one or more domains, select the 'Only objects in select domain(s)' option. In the text box that appears, enter one or more domains: <ul style="list-style-type: none"> • Enter the domain to replicate the objects within a <i>specific domain</i>. • Enter multiple domains separated by commas and/or use pattern matching to replicate objects within <i>multiple domains</i>. • Enter domains to exclude them from replication. (v10.0) <p>The field value allows pattern matching with the Python regular expression (RE) syntax so multiple domain names can be matched. The exception to the RE matching is the "{m,n}" repetitions qualifier may not be used.</p> <p>An example domain list value using RE is: <code>.*\example\.com</code> This matches both of these domains: <code>accounting.example.com</code>, <code>engineering.example.com</code>.</p> <div data-bbox="272 1094 1422 1434" style="border: 1px solid #ccc; padding: 10px; margin-bottom: 10px;"> <p>Scope <input type="radio"/> Entire source cluster (global) <input checked="" type="radio"/> Only objects in select domain(s)</p> <p>Include domains</p> <div style="border: 1px solid #ccc; padding: 2px; margin-bottom: 5px;"> 🔍 <input type="text" value=".*\example\.com"/> </div> <p>Exclude domains</p> <div style="border: 1px solid #ccc; padding: 2px; margin-bottom: 5px;"> 🗑️ <input type="text" value="private.example.com"/> </div> <p><input checked="" type="checkbox"/> Include objects without a domain</p> </div> <ul style="list-style-type: none"> • Inclu <p>– To replicate any unnamed objects that are not tenanted in any domain, enable the option.</p>
<p>Target Remote Cluster Name</p>	<p>The configuration setting for the target cluster (the <code>cluster.name</code> parameter in the <code>.cfg</code> file of the target cluster). configure the Gateway setting <code>allowSwarmAdminIP</code> when using Gateway as target. See Gateway Configuration</p>

Proxy or Host(s)	<p>The IP address(es) or host name(s) of either:</p> <ul style="list-style-type: none"> • One or more nodes in the target cluster. It is best to list all nodes in the target cluster, while swarm follows redirects it does not use those redirect targets for subsequent requests. • A reverse proxy host that routes to the target cluster. <p>To enter two or more node IP addresses, enter each address separated by a comma or spaces.</p>
Port	<p>Defaults to 80. Allows specifying a custom port for the remote cluster.</p>
Replication Mode	<p>Defaults to Direct POST. Choose replication via direct POST (recommended) or bidirectional GET. Switching modes does not require a feed restart. (v9.6)</p> <p>For best performance, choose direct POST replication, which can go through Gateway. GET replication is the legacy method, which may be needed for application compatibility or networking requirements.</p>
Threads	<p><i>Replication via direct POST.</i> The default replication speed (6 simultaneous threads) is best for same-sized clusters with minimal replication backlog. When processing backlog, the thread count is per source cluster volume. (v9.6)</p> <p>Reduce the threads to avoid overwhelming a smaller target cluster. For faster replication against a backlog, increase the threads temporarily, monitor bandwidth and cluster performance, as boosting the speed stresses both clusters.</p>
SSL Server	<p><i>Replication via direct POST.</i> Defaults to none. Enable Require trusted SSL if replicating over an untrusted network; Allow untrusted SSL is available but not intended for production systems. (v10.0)</p>
Remote Admin Name /Password	<p>Inherit from source cluster: Uncheck the enabled box, <i>if</i> the remote cluster user name is different from the source cluster name in the same realm. Then enter:</p> <p>User/Password credentials</p> <ul style="list-style-type: none"> • The administrative user name of the target cluster. • The administrative password of the target cluster.



Propagate Deletes

The legacy option **Propagate Deletes** is deprecated; it existed to allow setting a target cluster to preserve all deleted objects. This need is now covered by [Object Versioning](#); historical versions of deleted objects can be accessed to recover mistakenly deleted content. Versioning to the target cluster serving as the archive can be limited, to minimize space usage. (v11.1)

Note these restrictions and behaviors if an existing feed has this option specified:

- **With Versioning** – *Always* propagate deletes when using [Object Versioning](#) in the cluster.
- **Without Versioning** – The target cluster maintains deleted content carrying no verifiable indication of the deleted status if this option is disabled.

Using Feed Actions

Clicking on an existing feed in the Feeds list opens the **Feed Settings** page, with the existing settings populated. The Actions (gear) icon menu at the top right supports multiple feed actions, appropriate to the type of feed:

<p>Pause / Resume</p>	<p>Feed processing may occasionally need to be paused to perform system maintenance. Pause the search feed before stopping the Elasticsearch service in the search cluster when upgrading an Elasticsearch cluster. Return to the action menu and select the Resume action to resume feed processing after completing system maintenance.</p>
<p>Refresh</p>	<p>Object data is sent to the feed target in near real-time (NRT) as they are written or updated. Any objects unable to be processed immediately are retried each HP cycle until successful, at which point they are marked as complete and are not resent. Select the Refresh option from the feed action menu, which verifies and rehydrates all previously sent content to a remote cluster if a data loss failure occurs on the remote feed target and restoration from backup is not possible. This process takes some time, as it must revisit all objects in the cluster.</p>
<p>Delete</p>	<p>When a feed is deleted, it frees source cluster resources. This process does not affect the objects previously pushed to the remote target. Select the Delete option from the feed action menu and verify the intention to permanently delete the feed. The deleted feed is removed from the remaining cluster nodes within 60 seconds.</p>
<p>View feed table</p>	<p>Displays the SNMP Repository Dump for the selected node, for feed diagnostics (see below).</p>

Troubleshooting Feeds

- **Feed diagnostics** – Double-click a blocked feed to open the settings page, click the gear icon, and select **View feed table**, which displays the SNMP Repository Dump for the selected node to troubleshoot. (v2.0)



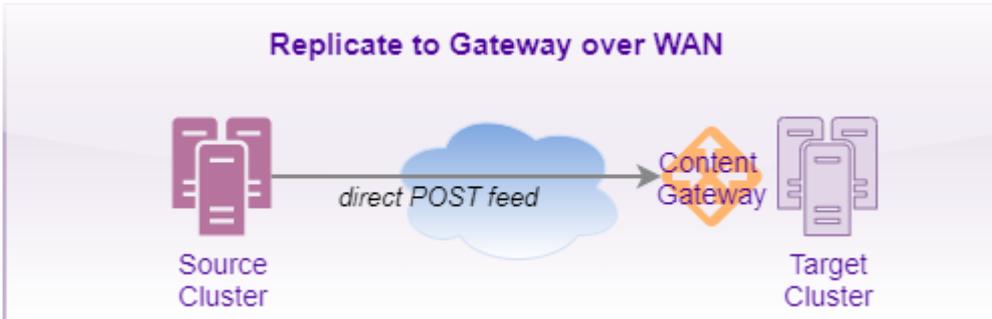
Review the **feedPluginState** status to identify the blockage.

Example: `feedPluginState blocked: Destination cluster onyx1 reports invalid request: Castor-System-Cluster value must refer to a remote cluster on RETRIEVE request`

- **Idle feeds** – A feed can appear to be idle with items still queued for processing. Plan for the fact that feed status reporting is a best-effort snapshot, not a low-latency or guaranteed transaction mechanism.
- **Feed prioritization** – Domain and bucket context objects are prioritized for *all* types of feeds; this improves usability when remote sites are initiated.
- **Retries for blocked feeds** – Blocked feeds are retried every 20 minutes, but if the definition for a blocked feed is changed, it triggers an immediate attempt with the new definition, which may clear the blockage. (v10.1)

Replication Feeds over Untrusted Networks

With Content Gateway 5.4 and higher, you can create replication feeds using direct POST replication, with Gateway proxying the target cluster.



Note

Gateway's `cluster.proxyIPList` setting is for use with legacy bidirectional GET replication. See [Managing Feeds](#).

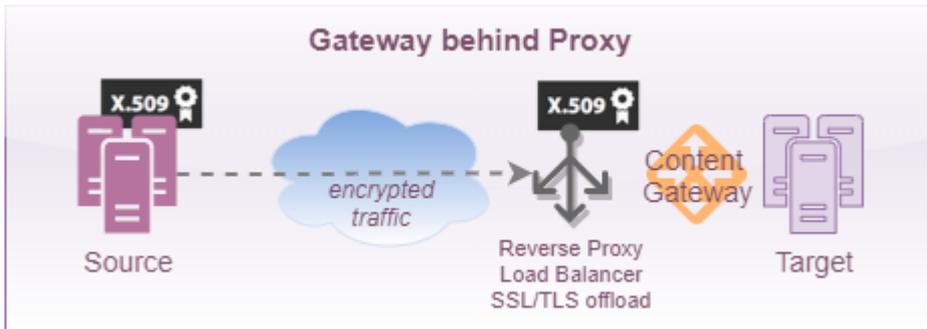
Add a trusted certificate and a third-party proxy to handle redirects if the source cluster needs to replicate feeds over an untrusted network securely with SSL/TLS encryption.

See [Adding a Trusted Certificate to Swarm](#) for more on creating a self-signed SSL certificate.

- [Note](#)
- [Load Balancer \(Offloader\)](#)
 - [Setting Up an Offloader](#)
 - [Important](#)
 - [Configuring the Feed](#)
- [Forward Proxy](#)
 - [Choosing a Forward Proxy Server](#)
 - [Re-configuring the Feed](#)

Load Balancer (Offloader)

Configure the source Swarm cluster and load balancer to use a trusted connection if using a load balancer for SSL/TLS offload (which is a type of reverse proxy):



Setting Up an Offloader

Important

The ports specified in the proxy configuration *must* match the bind ports specified in the [Gateway Configuration](#).

This example shows how to configure haproxy as an SSL offloader for Content Gateway on RHEL/CentOS 7.

1. Check the Content Gateway configuration and note which ports are being used for SCSP and S3. These ports *must* match in the offloader's setup.

/etc/caringo/cloudgateway/gateway.cfg

```
[scsp]
enabled = true
bindAddress = 0.0.0.0
bindPort = 8080
externalHTTPport = 443

[s3]
enabled = true
bindAddress = 0.0.0.0
bindPort = 8090

[cluster_admin]
enabled = true
bindAddress = 0.0.0.0
bindPort = 91
externalHTTPSport = 91
```

Forcing HTTPS

This configuration still provides HTTP access; to harden security and force HTTPS, change all of the `bindAddress` settings to 127.0.0.1.

2. Setup and install haproxy. This package is part of the EPEL repo.
3. Use the following haproxy configuration:

`/etc/haproxy/haproxy.cfg`

```

global
    log 127.0.0.1 local2
    chroot /var/lib/haproxy
    stats socket /var/lib/haproxy/stats mode 660 level admin
    stats timeout 30s
    user haproxy
    group haproxy
    daemon

    ca-base /etc/pki/tls/certs
    crt-base /etc/pki/tls/private

    ssl-default-bind-ciphers ECDH+AESGCM:DH+AESGCM:ECDH+AES256:DH+AES256:ECDH+AES128:DH+AES
    ssl-default-bind-options no-sslvs3
    maxconn 2048
    tune.ssl.default-dh-param 2048

defaults
    log global
    mode http
    option forwardfor
    option http-server-close
    option httplog
    option dontlognull
    timeout connect 5000
    timeout client 50000
    timeout server 50000

frontend www-http
    bind 0.0.0.0:80
    reqadd X-Forwarded-Proto:\ http
    reqadd X-Forwarded-Port:\ 80
    default_backend www-backend-scsp
    acl iss3 hdr_sub(Authorization) AWS
    acl iss3 url_reg [?&](AWSAccessKeyId|X-Amz-Credential)=
    use_backend www-backend-s3 if iss3

frontend www-https
    bind 0.0.0.0:443 ssl crt /etc/pki/tls/certs/selfsignedcert.pem
    reqadd X-Forwarded-Proto:\ https
    reqadd X-Forwarded-Port:\ 443
    default_backend www-backend-scsp
    acl iss3 hdr_sub(Authorization) AWS
    acl iss3 url_reg [?&](AWSAccessKeyId|X-Amz-Credential)=
    use_backend www-backend-s3 if iss3

frontend www-https-svc
    bind 0.0.0.0:91 ssl crt /etc/pki/tls/certs/selfsignedcert.pem
    reqadd X-Forwarded-Proto:\ https
    reqadd X-Forwarded-Port:\ 91
    default_backend www-backend-svc

backend www-backend-scsp
    #redirect scheme https if !{ ssl_fc } <--- Uncomment to force HTTPS
    server gw1 127.0.0.1:8080 check

backend www-backend-s3
    #redirect scheme https if !{ ssl_fc } <--- Uncomment to force HTTPS
    server gw1 127.0.0.1:8090 check

backend www-backend-svc
    # This rule rewrites CORS header to add the port number used on frontend
    http-request replace-value Access-Control-Allow-Origin (.*) \1:91
    redirect scheme https if !{ ssl_fc }
    server gw1 127.0.0.1:8091 check
    
```

4. Start haproxy.

```
systemctl restart haproxy
```

5. Edit the existing feed to enable SSL and point to the new endpoint (see next).

Configuring the Feed

Configure the Swarm replication feed to use the SSL server once it is configured.

1. In the Swarm UI, navigate to **Cluster > Feeds**.
2. Edit the affected replication feed.
3. Scroll to the **Target Remote Cluster** settings.
4. Update the **Proxy or Host(s)** and **Port** to point to the offloader.
5. Select **Replicate via direct POST** if the feed was configured to use the bidirectional GET mode.
6. Enable **Require trusted SSL for SSL Server**; **Allow untrusted SSL** is available but not intended for production systems.

Target Remote Cluster

Cluster Name

Proxy or Host(s) 1

Port 2

Replication Mode

Replicate via bidirectional GET (if needed)

Replicate via direct POST (recommended; supports Gateway)

Threads

SSL Server 3

None Require trusted SSL Allow untrusted SSL (not recommended)

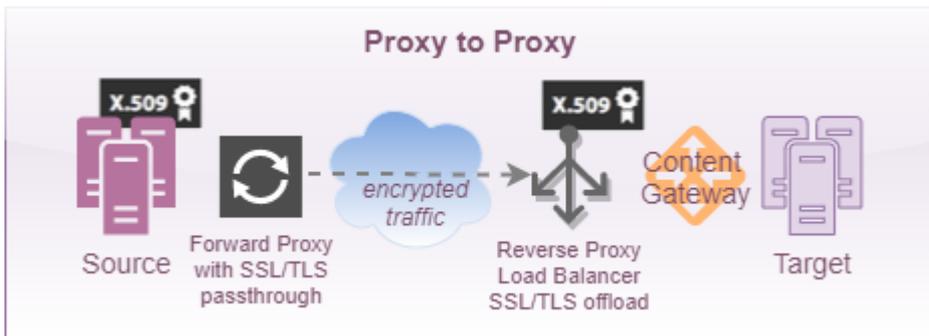
Local Cluster Forward Proxy

None Use proxy

7. Select **None** for **Local Cluster Forward Proxy**, unless using one (see *Forward Proxy*, below).

See [Managing Feeds](#).

Forward Proxy



Choosing a Forward Proxy Server

[HAProxy](#) works with a fixed back-end server list consisting of the distant Gateway front-end although it is not optimized to be a general purpose forward proxy.

Other alternatives:

- [stunnel](#) – for fixed endpoints
- [Squid](#) – for a general purpose forward proxy

With this server configuration, the forward proxy receives an HTTP request from the Swarm node and then tunnels a Swarm HTTPS request over the Internet to the other cluster, hitting the SSL/TLS offloader in front of Gateway. The data is encrypted by Swarm, and passes blindly through the forward proxy.

Re-configuring the Feed

Configure the Swarm replication feed to use the new outbound proxy once a forward proxy server is configured.

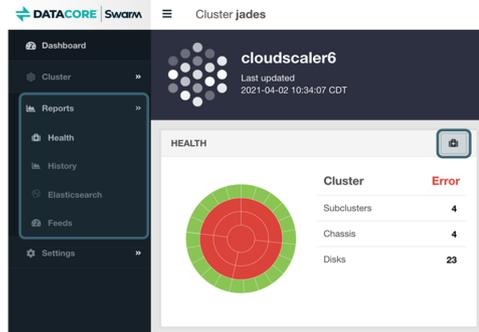
1. In the Swarm UI, navigate to **Cluster > Feeds**.
2. Edit the replication feed already configured to use an SSL Server.
3. Select **Use proxy** for **Local Cluster Forward Proxy**.
4. Enter the **Host** (a fully qualified domain name or IP address) and the **Port** for the proxy.
5. Enter the **Username** and **Password** for the forward proxy.

See [Managing Feeds](#).

Using Cluster Reports

The Reports section of the Swarm UI includes valuable real-time and historical views into the health and activity of a swarm cluster.

- [Health Report](#)
- [History Reports](#)
- [Elasticsearch Reports](#)
- [Feeds Reports](#)



Health Report

The Health Report provides both summary and detail information at the level of cluster, subcluster, chassis, and drive.

The sunburst graphic shows an interactive visualization of the cluster, with the cluster itself represented in the center, subclusters in the next concentric circle out, chassis in the next concentric circle, and drives on the outside. To see only the data for a particular subcluster or chassis, click on its wedge in the sunburst. All summary and detail data update to show only the selected component. To return to a higher level view, click the center of the sunburst. The component's ID (IP Address, drive name, etc.) and status display when moving the cursor over

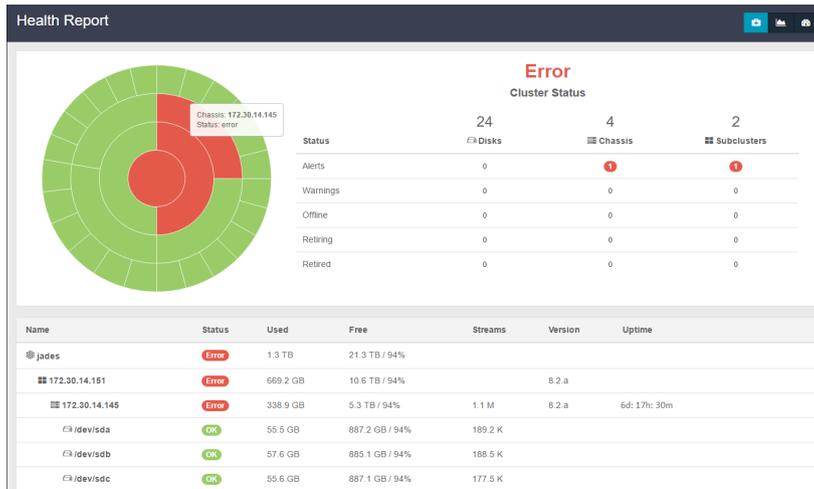
each wedge in the sunburst to make it easier to identify a component.

The status of each component is represented by the color of its wedge in the sunburst. Statuses include the following:

OK	The chassis or drive is working and there are no errors.
Alert Warning	The chassis or drive has experienced one or more errors. Cluster-level alerts often relate to space thresholds or network issues (unable to reach NTP/Gateway/ES/Metrics servers or other nodes).
Initializing	The short state after a chassis boots when it is reading cluster persisted settings and is not quite ready to accept requests.
Maintenance	The chassis has been shut down or rebooted by an administrator from either SNMP or the UI and should not be considered missing for recovery purposes. By default a chassis can be in a Maintenance state for 3 hours before it transitions to Offline and the cluster starts recovery of its content. Maintenance mode is not initialized when the power is manually cycled on the chassis outside of Swarm (either physically on the hardware or via a remote shutdown mechanism in an out-of-band management platform such as IPMI, Dell iDRAC, or HP iLO) or if there is a drive error; in both these instances recovery processes start for the chassis/drive unless recovery is suspended.
Mounting	The chassis is mounting one or more drives, including formatting the drive if it is new and reading all objects on the volume into the RAM index for faster access.
Offline	The chassis or drive was previously but is no longer present in the cluster.
Retiring	The chassis or drive is in the process of retiring, verifying all objects are fully protected elsewhere in the cluster and then removing them locally.
Retired	The chassis or drive has completed the retiring process and may be removed from the cluster.
Idle	The chassis or drive is in power-saving mode due during a period of configurable inactivity. (See Configuring Power Management.)

Subcluster and **Cluster** status is inherited from the chassis or drives contained within.

The data table below the sunburst displays more detailed information about the cluster, including the amount of used and free capacity and how many streams reside on the chassis/drive. Clicking on a subcluster row loads the **Subcluster** page. Clicking on a chassis row takes loads the **Chassis Details** page unless the chassis status is **Maintenance** or **Offline**.



History Reports

The History Reports include several charts that display daily usage and activity in the cluster over a rolling 30-day window. Mouse over the chart to see the exact values for all included data sets on any given day. Toggle the displayed data by clicking on each value in the legend at the bottom of each chart if there are multiple data sets included in the charts. Click the **Size** label to remove the data from the display, to see **Streams** on the **Storage Contents** chart.

The History Reports depend on [Swarm Historical Metrics](#) to provide historical data for the charts. The chart displays **Data unavailable** if Historical Metrics are not available or there is no data available within the last 30 days.

Important

Swarm generates CRITICAL log messages if [Historical Metrics](#) is misconfigured or if connection to the Elasticsearch cluster is lost.

Troubleshooting Data Unavailable

Charts show **Data unavailable** when it cannot obtain Swarm metrics data to display. These are possible causes:

- Metrics data has not been recorded or is missing (ES returns an "index not found" exception)
- `metrics.targets` is not configured
- The Metrics Curator is not configured properly, or the crond service is not running (see [Installing Swarm Metrics](#))
- The Service Proxy cannot contact Elasticsearch (see [Service Proxy](#))
- Elasticsearch is not configured to enable `http.cors` (see [Configuring Elasticsearch](#))

Run the following command (provided in [Installing Swarm Metrics](#)) to verify metrics are configured correctly:

```
curl "http://ESHOST:9200/metrics-CLUSTERNAME-scsp-all/metrics/_search?pretty=true"
```

Provide the contents of the browser's console to assist DataCore Support with resolution (UIS-494)

- Chrome: F12; Ctrl + Shift + J (PC); Cmd + Option + J (Mac)
- Firefox: F12; Action button > More tools > Web Developer Tools or Browser Console
- Edge: F12; Action button > More tools > Developer Tools

Storage Contents

The Storage Contents chart displays the total amount of used capacity in the cluster over time as well as the total stream count (including replicas and erasure coding segments).

STORAGE CONTENTS



Usage

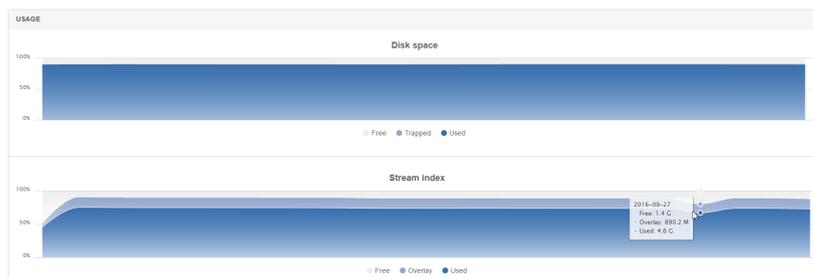
Note

Historical usage charts may show artificial bumps in usage when adding or removing a large percentage of drives within a single day.

The Usage charts display percentages of **Disk space** and **Stream index** (memory):

Disk space – The amount of free, trapped and used drive space as a percent of the total available over time.

Stream index – The amount of free, [overlay](#), and used RAM index space as a percent of the total available over time.

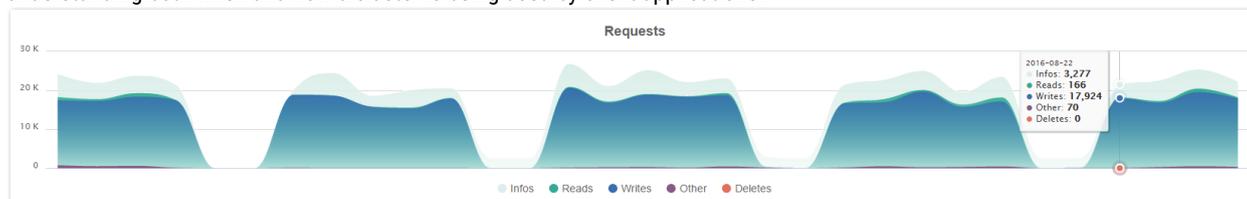


Network Traffic

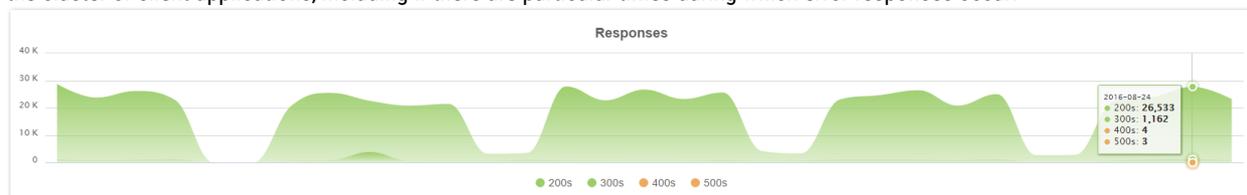
The Network Traffic graphs display **Requests**, **Responses**, and **Internal Requests** (inter-cluster activity).

Requests – The count of each SCSP method type in incoming client requests to the cluster over time. SCSP Method types are: [INFO](#), [READ](#), Writes (sum of [WRITE](#), [UPDATE](#) and [APPEND](#)), [DELETE](#), and Other (sum of [COPY](#) and [Search queries](#)). This information is useful in

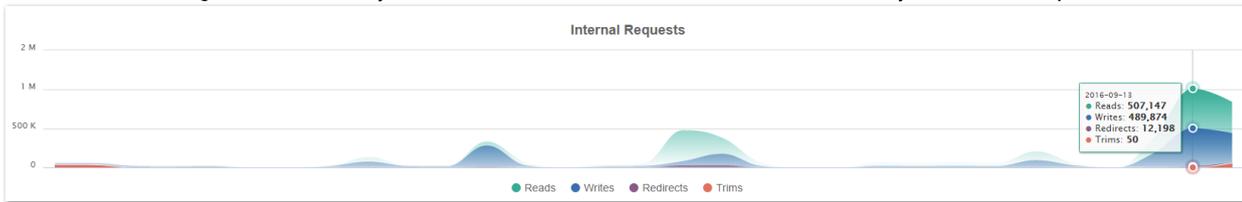
understanding both when and how a cluster is being used by client applications.



Responses – The count of [HTTP response codes](#) returned to clients by the storage cluster over time. This data is helpful in identifying problems in the cluster or client applications, including if there are particular times during which error responses occur.



Internal Requests— The count of various internal, cluster initiated activities between nodes in the cluster over time. This information is helpful in understanding how much data movement is happening in the cluster as hardware is added, removed, retired, etc. Spikes in activity within the cluster not correlating with client activity are often associated with either a failed drive recovery or an admin requested retire.



Elasticsearch Reports

Research an ES cluster status on the **Elasticsearch Reports** page if the **Elasticsearch** panel on the Dashboard shows a problem. These reports generate on demand and allow drilling into details spanning the ES nodes, thread pools, indices, and shards. (v2.0)

Important

Opening the **Elasticsearch Reports** page requires generation of a lot of status data; allow time for the page to display.

For details on the columns that are reported, see the relevant Elasticsearch Reference: [version 2.3](#) or [version 5.6](#).

Section	Setting	Notes
RESOURCES Node details	<ul style="list-style-type: none"> name ip uptime master cpu disk avail memory size tripped breaker file desc current heap max heap percent ram percent indexing delete total indexing index total search query total 	<p>Shows the ES cluster topology.</p> <p>For seeing where a node resides and to check performance stats, focus on these columns:</p> <ul style="list-style-type: none"> <i>ip</i> <i>cpu</i> <i>tripped breaker</i> <div style="border: 1px solid #ccc; padding: 5px; margin: 10px 0;"> <p>Important</p> <p>The tripped breaker field signals trouble. Contact DataCore Support if the status is red.</p> </div> <ul style="list-style-type: none"> <i>heap percent</i> <i>ram percent</i> <p>Other columns are more helpful when looking at larger clusters, such as determining how many master-eligible nodes are available:</p> <ul style="list-style-type: none"> <i>master</i> <i>name</i>

RESOURCES Thread pool details	<ul style="list-style-type: none"> • name • ip • bulk rejected • flush rejected • force_merge rejected • generic rejected • get rejected • index rejected • refresh rejected • search rejected • warmer rejected 	Shows ES cluster-wide thread pool statistics per node. The <code>rejected</code> statistics are returned for <i>all</i> thread pools.
INDICES	<ul style="list-style-type: none"> • index • health • status • docs count • docs deleted • pri • pri store size • rep • store size 	Provides low-level information about the segments in the shards of an index. <div data-bbox="603 684 1484 856" style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>Swarm Metrics generates large numbers of indices.</p> </div> <p><i>docs.count</i> – The number of non-deleted documents stored in this segment. These are Lucene documents, so the count includes hidden documents (such as from nested types).</p> <p><i>docs.deleted</i> – The number of deleted documents stored in this segment. The space for these documents is reclaimed when this segment is merged</p>
SHARDS	<ul style="list-style-type: none"> • index • node • ip • docs • prirep • shard • state • store 	The detailed view of what nodes contain which shards. It tells if it is a primary or replica, the number of docs, the bytes consumed on disk, and the node where it is located. <p><i>prirep</i> – Whether this segment belongs to a primary or replica shard.</p>

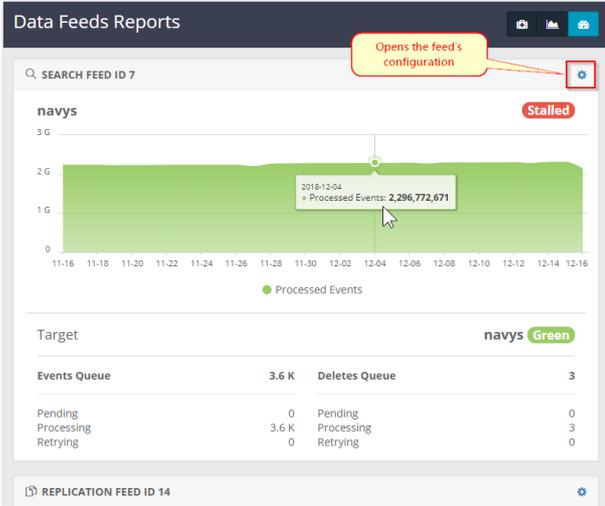
Feeds Reports

The Data Feeds Reports show the number of processed events for each configured search or replication feed over time, providing insight into how busy each feed is. Status markers alert to problems with the feed.



Tip

For quick access to the configuration details for a feed, click the gear icon in the top right of the chart.



Swarm UI Essentials

- [Accessing the UI](#)
- [Navigating the UI](#)
- [Resources](#)
- [Rebranding the UI](#)

Accessing the UI

The URL used to access the Storage UI depends on where it is installed. See [Installing the Storage UI](#) to determine the starting URL.

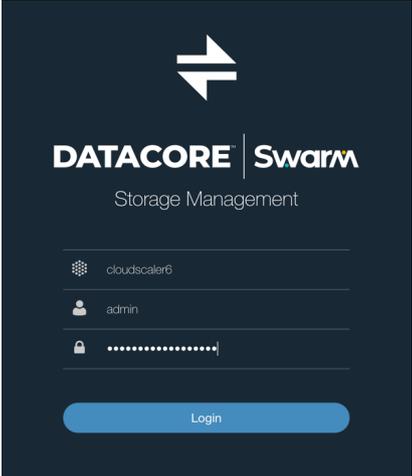


Port 91

The Swarm UI defaults to port 91 (`http://CSN·HOST:91/_admin/storage`). Use the configured port if the `bindPort` in `[cluster_admin]` was modified for the [Service Proxy configuration](#).

A prompt to log in appears upon reaching the UI. User logins for the UIs are not Swarm-managed but rather LDAP, PAM, or SAML, as configured by the Gateway IDSYS (see [Gateway Identity System](#)).

How the login screen appears depends on whether single sign-on is enabled:

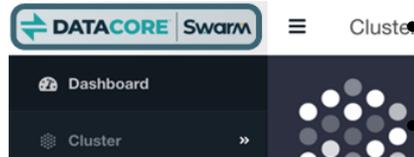
Without single sign-on	With single sign-on
 <p>Host – <i>Read-only</i>. The host name or IP address of the Swarm storage cluster to be viewed.</p>	 <p>User Name – The user name to be used to sign in through an existing third-party account, such as through OneLogin, Okta, or Google.</p> <p>Users are redirected to the other site to enter the password, and then are redirected back to the Swarm UI when clicking Login with SSO. The login typically expires in one day.</p> <p>For SSO setup, see Enabling SSO with SAML. (v3.0)</p>

Navigating the UI

Chose which Swarm UI to use (Storage or [Content](#)) upon logging in:

Click the top of the site map (the DataCore Swarm logo) to return to this page at any time to navigate to the other UI:

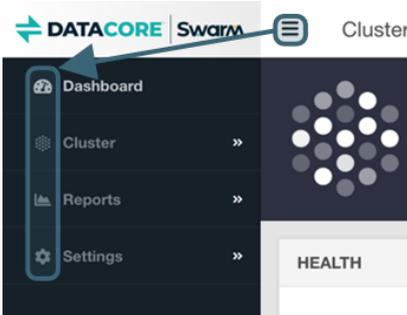
The primary navigation for the Swarm UI is in the left side navigation pane, which includes 3 major sections:



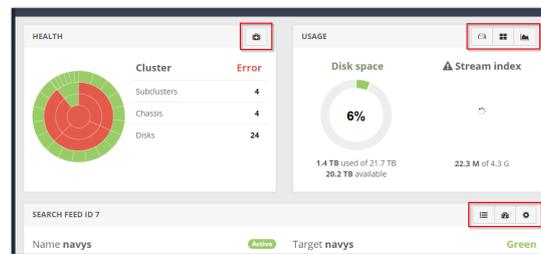
Cluster: provides access to the details of the cluster's hardware (physical chassis), subclusters (if any), and feeds (search and replication).
Reports: includes valuable real time and historical views in to the health of and activity in the cluster.

Settings: displays cluster-wide settings and license information.

The left pane can be opened and collapsed using the stacked bar icon next to the DataCore Swarm logo.

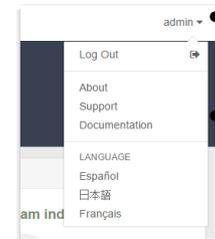


Throughout the application, quick access links to relevant reports and pages can be accessed through icons on each page. The Health section of the Dashboard includes a link to the more detailed Health Report.



Resources

Located at the top right of the Storage UI is the account name, which drops down a menu of resources:



Log Out – ends the current session
About – reports the version of the software in use with links to the end user license

agreement and third-party licenses

- **Support** – opens the [Support site](#) (requires an account for full access)
- **Documentation** – opens searchable online help
- **Language** - allows selection of an alternate display language for the user interface

Localization

Selecting an alternate language from the resource menu localizes most but not *all* text. Log messages, error messages, setting descriptions, and some chart labels are not localized as they come from sources outside the UI. Setting a language preference through the browser may return more accurate translation results.

Rebranding the UI

Use the '**custom**' folder that comes with Swarm UI and overwrite the included files to update the UI look and feel for the organization.

1. Place all custom files, SVGs, and style sheets into this '**custom**' directory.
2. Replace the images provided or create new images and update the style sheet to reflect any naming changes.
3. Uncomment **custom.css** for the styling updates to work.

Create the **custom** folder at the top level of the installed JavaScript files, alongside such folders as **app**, **fonts**, **scripts**, **styles** if the folder does not exist.

help-UIS

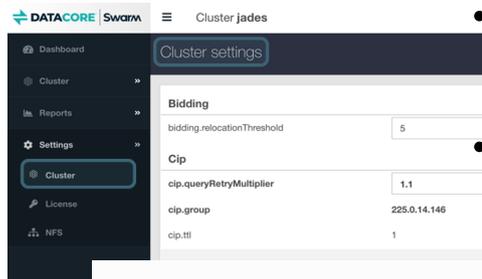
Using Cluster Settings

The **Settings** menu includes cluster-wide settings, license information, and optional NFS configuration.

- [Cluster Settings](#)
 - [Note](#)
- [License Settings](#)
 - [Note](#)
- [NFS Settings](#)

Cluster Settings

The **Settings > Cluster** menu option opens a page for managing the configuration of a storage cluster. What appears here and whether it is editable depends on the scope, dynamic state, and is using Platform Server.



- **Are these all settings?** See the [Settings Reference](#), which organizes the settings by scope: *Cluster + Chassis (Node)*. Cluster-wide settings are managed here under **Cluster Settings**. Node-specific settings are not a part of cluster configuration; if needed, DataCore Support can help access them using SNMP or the **Chassis Details** page **Advanced** tab's API tools. (v2.2)
- **What are "advanced settings"?** Advanced settings are those in force for a cluster but are not published in the [Settings Reference](#). Clicking **Show advanced settings** allows viewing and changing the values for these hidden settings. All are dynamic (see next). (v2.2)

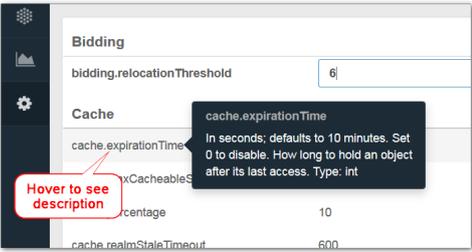
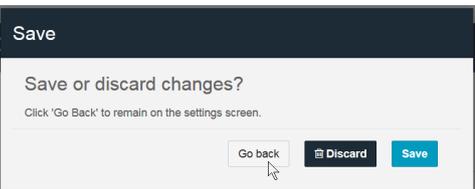
Caution

Do not change advanced settings except as directed by DataCore Support.



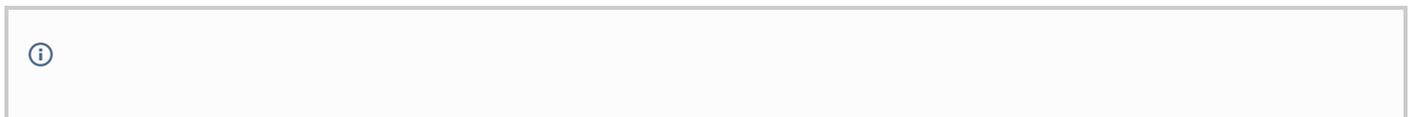
- **Which are dynamic settings?** See the [Settings Reference](#), which shows the **SNMP name** for each setting that can be changed on a running cluster.
- *Without* Platform Server running, every setting can be changed on demand, except for the non-dynamic settings, which appear as read-only.
- *With* Platform Server running, every setting can be changed on demand, but non-dynamic settings display an icon indicating a reboot is needed.

Here is how to make best use of the **Cluster Settings** page:

<p>Descriptions</p>	<p>Mouse over the setting name to see the setting description; refer to the Settings Reference for complete details.</p> 
<p>Writable values</p>	<p>Settings that can be updated on a running cluster display with an edit box or radio toggle to allow for quick editing.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>The cluster configuration file must be updated (on the CSN or wherever it resides) and all storage chassis must be rebooted for new settings to be applied to edit settings not supporting dynamic updates (not persisted settings).</p> </div>
<p>Custom values</p>	<p>All customized settings appear in bold for easy identification. This does not apply to settings with no default.</p> 
<p>Changing values</p>	<p>Changing any editable settings enables the Save button in the upper right corner. Any in process changes can be returned to the previous saved value by using the Revert button. The page prompts the user to decide to Save, Discard or Go Back if navigating away from the page with unsaved changes:</p>  <p>Settings changes propagate in the cluster within 60 seconds once clicking Save.</p>

License Settings

The license information for the license currently in use by the cluster displays on the **License Settings** page. Scroll down to see the date it was last updated.



Note

The License Settings page requires [Platform Server](#) to update the license from the Swarm UI; otherwise, it is read-only.

Site license

Company	DataCore
Cluster Description	Lab license Germany
Address	Various
City	Locations
State/province	
Zip/postal code	
Country	EMEA
Licensed capacity	250 TB
Expiration date	2022-01-08
Serial number	20210108030016-32193
Erasure coding	true
Full metadata search	true

NFS Settings

SwarmFS is an optional component to support network file sharing. See [SwarmFS Export Configuration](#).

- [Adding a Trusted Certificate to Swarm](#)

Adding a Trusted Certificate to Swarm

- [Certificate Essentials](#)
 - [Note](#)
- [Making a Self-Signed SSL Certificate](#)
 - [X.509 Required](#)
- [Uploading Certificates into Swarm](#)
 - [Tip](#)

Certificate Essentials

An X.509 security certificate (also called an "SSL Certificate") needs to be added so Swarm can be trusted for a secure connection if the Swarm site collects or transmits personally identifiable information or otherwise needs to protect traffic. TLS (Transport Layer Security) or SSL (Secure Sockets Layer) security is made up of two parts:

1. **Encryption** – data is made unreadable (using an *encryption key*) and then sent over an HTTPS connection (SSL), and it can read by a client with the needed *key*.
2. **Identification** – transmission is certified (with a *security certificate*) as coming from the authentic (*trusted*) site.

There are two options for certificates:

- Pay a trusted CA (*Certificate Authority*, such as Verisign) to approve (*sign*) a certificate. This is needed for e-commerce.
- Create a *self-signed* certificate.

Both certificate types encrypt data to create a secure website third-parties cannot read.



Note

Most browsers check whether an HTTPS connection is signed by a recognized CA. It can be flagged as potentially risky even though it is secure if the connection is self-signed.

Making a Self-Signed SSL Certificate



X.509 Required

Swarm requires X.509 formatted certificates, which is a public key infrastructure standard SSL and TLS adhere to for key and certificate management.

There are many ways and tools for creating trusted certificates. Here is one way for making a self-signed certificate to add to a proxy for use in front of a Swarm cluster:

1. Set up a secure (root-only access) directory for holding the private key and certificate files.
2. Generate a unique private key (KEY).

```
$ openssl genrsa -out mydomain.key 2048
```

File contents start with: -----BEGIN RSA PRIVATE KEY-----

3. Generate a Certificate Signing Request (CSR).

```
$ openssl req -new -key mydomain.key -out mydomain.csr
```

File contents start with: -----BEGIN CERTIFICATE REQUEST-----

4. Create a self-signed certificate (CRT), filling out the `openssl` prompts appropriately (most importantly, the `Common Name`, which may be a domain name or public IP address).

```
$ openssl x509 -req -days 365 -in mydomain.csr -signkey mydomain.key -out mydomain.crt
```

File contents start with: -----BEGIN CERTIFICATE-----

S3 wildcards

Create a wildcard SSL certificate if using S3. Run the command again, but use a wildcard: `*.DOMAIN` when prompted for **Common Name**.

1. Concatenate the `.crt` and `.key` files (in that order) into a `mydomain.pem` file.

```
$ cat mydomain.crt mydomain.key > /etc/pki/tls/certs/mydomain.pem
```

PEM file gains *multiple* sections: -----BEGIN CERTIFICATE----- and -----BEGIN RSA PRIVATE KEY-----
Repeat this step to create another `.pem` file if a wildcard cert was created

```
$ cat mydomain-wildcard.crt mydomain-wildcard.key > /etc/pki/tls/certs/mydomain-wildcard.pem
```

2. Specify the PEM in the configuration file of the proxy, such as HAProxy.

```
$ vi /etc/haproxy/haproxy.cfg
```

- **Locate:** `bind 0.0.0.0:443 ssl crt /etc/pki/tls/certs/selfsignedcert.pem`
- **Update to:** `bind 0.0.0.0:443 ssl crt /etc/pki/tls/certs/mydomain.pem crt /etc/pki/tls/certs/mydomain-wildcard.pem`

3. Restart the proxy.

```
$ systemctl restart haproxy
```

Uploading Certificates into Swarm

The trusted certificate (public key) needs to be uploaded to Swarm, which is performed by inserting it into Swarm's settings to protect Swarm traffic over untrusted networks. The `startup.certificates` setting holds all certificates, formatted as a single line.

Tip

Multiple certificates can be concatenated into one.

Platform only – Upload a certificate directly in the [Cluster Settings page](#) of the Swarm UI if running Platform Server 10.0 or higher. The UI handles the conversion of the certificate file to the single-line formatted required by Swarm. (v2.1)

No Platform – Certificates need to be prepared and uploaded by hand if not using Platform Server. Activating the certificate requires a reboot because it requires editing of the configuration file.

1. Modify PEM certificate(s) so the key is a single line, with all carriage returns replaced with the newline character: `\n`.
The following `awk` command converts a PEM file into the needed string:

```
awk 'NF {sub(/\r/, ""); printf "%s\\n",$0;}' mydomain.pem
```

2. Edit the Swarm configuration file (node.cfg, cluster.cfg).
3. Insert the certificates setting at the end of the file:

```
[startup]
certificates=
```

4. Directly after the equal sign, paste in the single-line string (shown here abridged), which contains all needed newline characters (\n):

```
[startup]
certificates=-----BEGIN CERTIFICATE----- {snip} -----END CERTIFICATE-----
```

5. Reboot the node. The value provided may not be a valid x.509 public certificate if Swarm does not boot, so check the formatting.

Swarm appends the key value to `/etc/ssl/certs/ca-certificates.crt` when it reboots.

Health Data to Support

Health data reporting allows DataCore to monitor your cluster and step in quickly when issues arise. Your cluster's generated health report data is viewable in the Swarm UI; this is the "health check" that your Swarm cluster automatically sends to Support.

Best practice

Monitoring this health information enables DataCore to provide technical support proactively, so it is best practice to participate in this support service. To include Gateway version and feature reporting, add the needed authorization the [storage_cluster] section of [Gateway Configuration](#). (v6.0)

Contact DataCore Support about disabling health reporting if data sharing needs to be disabled altogether.

- [Best practice](#)
- [Accessing Your Report](#)
 - [Deprecated](#)
 - [Troubleshooting](#)
- [Proxy for Health Reports](#)

Accessing Your Report

To see the health data and what information DataCore receives from your site, you can view the raw report in your browser: Navigate to **Cluster > Hardware**, double-click on hardware to open the **Chassis Details**, then open **Advanced, Health Data**. (v10.0)

Deprecated

The [Legacy Admin Console \(port 90\)](#) is still available but has been replaced by the [Swarm Storage UI](#). (v10.0)

Use this URL to view health reports there:

<host>:90/health_report

Troubleshooting

If you have problems accessing the health report, check the cluster network connectivity to the Internet and verify the cluster name is populated, there is a cluster settings file, and the `support.uri` parameter has not been disabled. To receive proactive support, use the default value "https://healthreport.caringo.com:443/castor/report".

Proxy for Health Reports

If you need to specify a proxy specific to health reports, add the following proxy settings as appropriate to your Swarm configuration. Swarm accepts a proxy username and password to allow health reports to be sent via a password-enabled proxy. (v9.2)

Setting	Notes
---------	-------

support.proxyUri	Proxy URI, which may but need not include http(s)://.
support.noProxy	Comma-separated list of domain names or IP addresses for which HTTP/S proxy should not be used. Do not include http(s):// or port numbers. Wildcards are allowed.
support.proxyUsername	Proxy authentication username.
support.proxyPassword	Proxy authentication password.

Viewing and Managing the Cluster

- [Streams versus Objects](#)
- [Dashboard](#)
 - [Empty dashboard](#)
 - [Tip](#)
 - [Note](#)
 - [Data delays](#)
- [Hardware](#)
- [Restarting or Shutting Down the Cluster](#)
 - [Admin only](#)
- [Suspending or Enabling Disk Recovery](#)
- [Subclusters](#)

The **Cluster** section provides access to the details of a cluster's **Hardware** (physical or virtual machines, or chassis), **Subclusters**, and **Feeds** (search and replication).

i Streams versus Objects

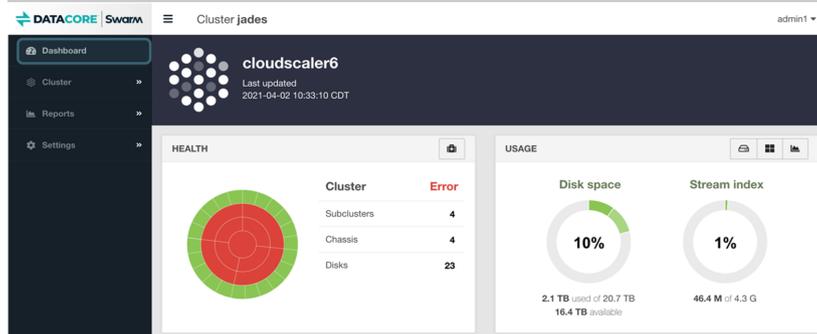
Streams are not objects. One object can involve *many* streams. **Objects** are the files uploaded to and download from Swarm, and they are indexed for searching and listing. **Streams** are all underlying data components Swarm manages internally, such as erasure-coded segments, replicas, historical versions, and chunks of multipart uploads. The Swarm UI reports on streams when it is surfacing Swarm's internal view of hardware usage.

Dashboard

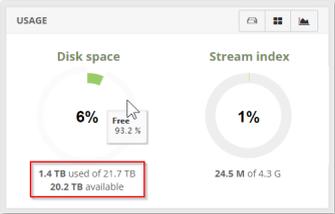
The primary initial view of the cluster is offered through the **Dashboard**. The Dashboard presents real-time visual monitoring, alerting, and history across the cluster's usage and activity. The host name and last refresh time for the page are shown at the top. The page auto-refreshes every 60 seconds.

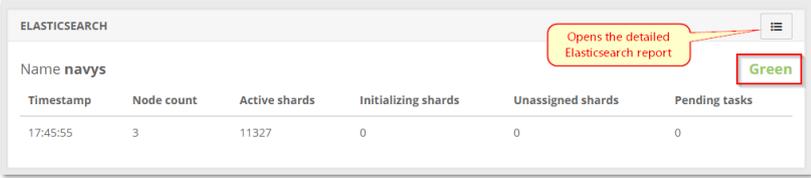
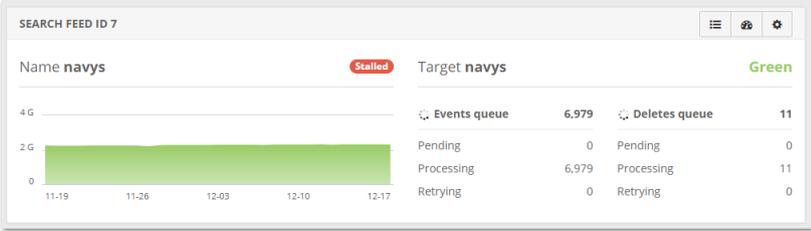
i Empty dashboard

The top of the page shows a Swarm watermark background if the cluster does *not* have an active license. Attempt to log out and log in again if the license is active but the data is still missing.

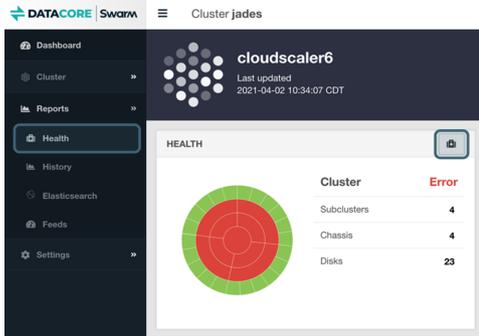


The Dashboard provides quick reference to these key indicators for the cluster:

<p>Health</p>	<p>Cluster</p> <p>Overall cluster health as a red, yellow, green status based on the status of the cluster's chassis and disks. Drill down to the Health Report for additional details.</p> <div data-bbox="336 354 1484 562" style="border: 1px solid #ccc; padding: 10px; margin-bottom: 10px;"> <p>i Tip</p> <p>Hover over sections to read the values in popup text if unable to distinguish among those colors.</p> </div> <div data-bbox="336 646 1484 854" style="border: 1px solid #ccc; padding: 10px;"> <p>i Note</p> <p>Per-object feed errors (for replication or search indexing) do not generate critical log messages and so do not change the node state to 'error'. Check the Feeds section of the Dashboard for blocked feeds.</p> </div>
<p>Usage</p>	<p>Disk space</p> <p>% of Disk space usage in the cluster, including free space, used space and trapped space. The Disk space pie chart turns orange if the amount of used capacity exceeds 80% and then red if the capacity exceeds 90%. Mouse over each segment in the pie chart for more information.</p> <p>Stream index</p> <p>% of Stream index usage in the cluster, including how much is used by the overlay index if any. The Stream index pie chart turns orange if the amount of used RAM exceeds 80%, and then red if the capacity exceeds 90%. Mouse over each segment in the pie chart for more information.</p> <div data-bbox="336 1325 671 1539" style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;">  <p>USAGE</p> <div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Disk space</p> <p>6% Free: 93.2%</p> <p style="border: 1px solid red; padding: 2px; font-size: 0.8em;">1.4 TB used of 21.7 TB 20.2 TB available</p> </div> <div style="text-align: center;"> <p>Stream index</p> <p>1%</p> <p>24.5 M of 4.3 G</p> </div> </div> </div>

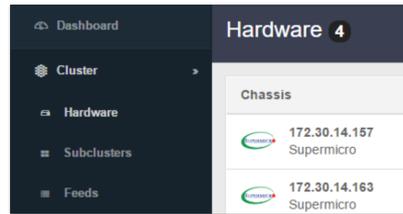
<p>Elasticsearch</p>	<p>Statistics</p> <p>The Elasticsearch panel provides a snapshot of the ES cluster's status and essential statistics. For details, click the icon for the full report.</p> 
<p>Feeds</p>	<p><i>One pane per feed</i></p> <p>Status for each feed, both search and replication, including current feed status, a trended graph of feed event processing over time, and a count by stage (<i>Pending, Processing, Retrying</i>) of any queued events and deletes. An animated spinning icon displays to indicate the feed is currently processing if queue for events or deletes is greater than zero.</p> <div data-bbox="336 793 1484 1039" style="border: 1px solid #ccc; padding: 10px;"> <p>Data delays</p> <p>It may take over 24 hours before the processing trend is visible in the graph.</p> <p>Spinners appear when the UI is waiting on feeds still processing data, but they also appear when the feed itself is busy.</p> </div> 
<p>Network Traffic</p>	<p>Request</p> <p>The count of each SCSP method type in incoming client requests over a rolling 30 day window.</p> <p>Responses</p> <p>The count of HTTP response codes returned to clients by the storage cluster over a rolling 30 day window.</p> 

The collapsible global menu pane provides navigation to more detailed information and reports. Use the icons on each section of the dashboard to drill down in to details for that section. Clicking the medical bag icon in the Dashboard's **Health** section opens the same page as selecting **Reports > Health** from the menu:



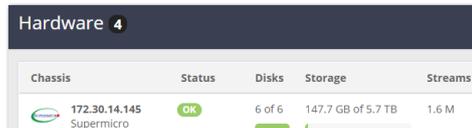
Hardware

The Hardware report includes a summary view of each server chassis in the storage cluster and the current state, including the number of chassis disks online, the used capacity, stream count, up-time, Swarm storage software version, and the subcluster to which the chassis belongs, if any.



Click on any row to drill down in to the Chassis Details page for that particular chassis (physical or virtual machine):

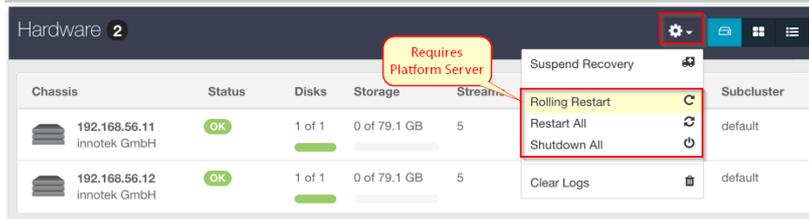
Restarting or Shutting Down the Cluster



The settings gear icon at the top of the page allows restarting or shutting down the entire storage cluster, as well as the ability to clear logs.

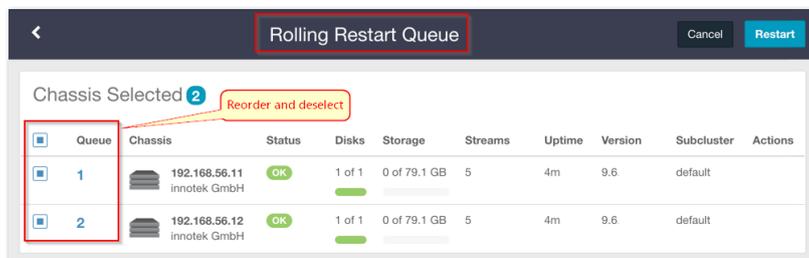
Admin only

These cluster-wide actions require authentication as an administrator.



Perform a Rolling Restart of the cluster with full control over the restart queue, to reorder and cancel individual chassis restarts if Platform Server is installed. (v2.0)

Suspending or Enabling Disk Recovery



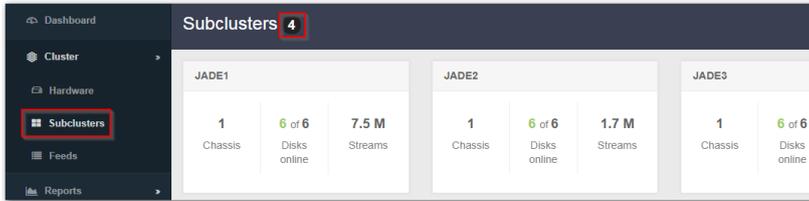
Administrators can suspend an in process disk recovery using the **Suspend Recovery** option under the settings gear icon.

Enable a previously suspended recovery using either the **Enable Disk Recovery** button in the banner message or the **Enable Recovery** under the settings gear icon.



Subclusters

The **Subclusters** page rolls up the information on all subclusters present in a cluster, including chassis and disk counts and number of streams.



Subclusters 4		
JADE1		
1 Chassis	6 of 6 Disks online	7.5 M Streams
JADE2		
1 Chassis	6 of 6 Disks online	1.7 M Streams
JADE3		
1 Chassis	6 of 6 Disks online	

To dynamically change a subcluster assignment, navigate to **Hardware > Chassis Details** and click the **Settings** tab. (Subclusters can be configured from the CSN (cluster.cfg) or in the node's configuration file (node.cfg), but these require a cluster reboot to take effect.)



Cluster settings	
Node	
node.archiveMode	<input type="radio"/> true <input checked="" type="radio"/> false
node.subcluster	jade1
Policy	

Legacy Admin Console (port 90)

 **Deprecated**

The Legacy Admin Console (port 90) is still available but has been replaced by the [Swarm Storage UI](#). (v10.0)

- [Managing Chassis and Drives - Legacy Admin Console](#)
- [Using Cluster Settings - Legacy Admin Console](#)
- [Viewing and Managing the Cluster - Legacy Admin Console](#)
- [UI Essentials - Legacy Admin Console](#)
- [Managing Domains - Legacy Admin Console](#)
- [Viewing and Editing Feeds - Legacy Admin Console](#)
- [Identifying the Primary Search Feed](#)

Managing Chassis and Drives - Legacy Admin Console

Deprecated

The [Legacy Admin Console \(port 90\)](#) is still available but has been replaced by the [Swarm Storage UI](#). (v10.0)

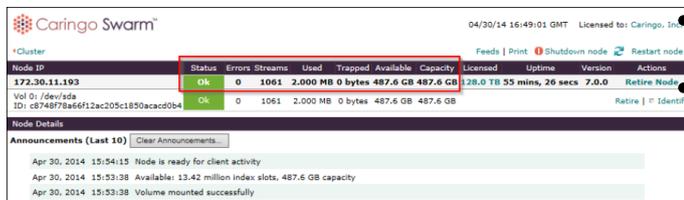
- [Viewing the Node Status Page](#)
- [Shutting Down or Restarting a Node](#)
- [Identifying a Drive](#)
- [Retiring a Drive](#)
- [Errors and Announcements](#)
- [Node Status Reporting](#)

Viewing the Node Status Page

Click the IP address on the left side of the Swarm Admin Console to view the status of a node. Expand a subcluster node name to display IP addresses and then click an IP address to display the node information if the cluster is configured to use subclusters.

See [Finding Nodes in the Cluster](#) to find a particular node.

The top row of the Node Status page provides summary information about the node and the associated volumes, such as up-time and storage usage statistics:



Streams: Counts the total number of managed data components (such as replicas and segments), not logical objects (such as video files).

Trapped: Calculates the space pending reclamation by the Swarm defragmentation process. This process is controlled by several Swarm parameters (see the [Settings Reference](#)).

Note

The node status page automatically refreshes every 30 seconds.

Shutting Down or Restarting a Node

Click **Shutdown Node** or **Restart Node** in the Swarm Admin Console to shut down or restart a node.

A node shutdown or rebooted by an Administrator appears with a Maintenance state on other nodes in the cluster.

See [Finding Nodes in the Cluster](#).

Identifying a Drive

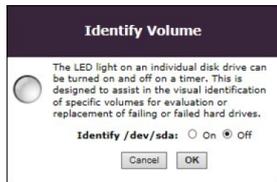
Identify one or all volumes on a node using the links on the right side of the Swarm Admin Console under **Restart Node**.

The **Identify** function allows selection of a particular volume and enable the corresponding LED drive light, which can be helpful in identifying a failed or failing drive. Select the targeted volume and the amount of time the light is enabled.

On the **Node Status** page, an **Identify** light displays next to the targeted volume for easy identification.

See [Drive Identification Plugin](#) for how to enable the drive light.

Swarm reverts to a default process to flash the light if a hardware-specific API is not used.



Retiring a Drive

Retire one or all volumes on a node using the links on the right side of the Swarm Admin Console under **Restart Node**.

On occasion, replacing Swarm volumes is required for regular maintenance or to upgrade the cluster nodes with higher capacity drives. Best practice is to retire volumes one at a time if multiple volumes need to be replaced across multiple Swarm nodes. Either choose a minimally disruptive retire limited to the volume(s) being retired, or an accelerated retire using all nodes in the cluster to replicate objects on the retiring volume(s) as quickly as possible when initiating a retire. Note the cluster-wide retire may impact performance as it does put additional load on the cluster.

Clicking **Retire Node** retires all volumes on the node, at the same time. Clicking **Retire** next to a volume retires the selected volume. A volume is also retired automatically if a configurable number of errors occur.

See [Retiring Volumes](#).

Verify the cluster meets the following conditions before retiring a node or volume:

- Has enough **capacity** for the objects on the retiring node to replicate elsewhere.
- Has enough **unique nodes** to replicate the objects with one replica on any given node.



Note

Retire succeeds if objects can be replicated elsewhere in the cluster. The Retire action does not remove an object until it can guarantee at least two replicas exist in the cluster or the existing number of replicas matches the **policy.replicas min** parameter value.

A retiring node or volume accepts no new or updated objects. Retiring a node or volume means all objects, including replicas, are copied to other nodes in the cluster.

On the Swarm Admin Console's Node Status page, the Node Operations section includes a **Retire Rate** tracking the number of objects per hour removed from a retiring volume. The SNMP MIB includes this same value in the **retireRatePerHour** MIB entry. The value is 0 if no volumes on the node are retiring.

The node or volume's state changes to **Retired** and Swarm no longer uses the node or volume after all objects are copied. At this point, remove and repair the volume or discard it.

Errors and Announcements

The last 10 errors and announcements appear on the **Node Status** page. The page is blank if there are no errors or announcements. The error count in the node summary grid corresponds to the list of errors in the error section.

Tip

Control how long uncleared error messages continue to appear in the error table by configuring the Swarm setting [console.messageExpirationSeconds](#), which defaults to two weeks.

Messages display in the node status area if removing or inserting a drive into a running node. This feature, referred to as [hot plugging](#) (adding a new drive) or [hot swapping](#) (replacing a failed drive), allows removal of failed drives for analysis or to add storage capacity to a node at any time.

The following message appears when adding a volume:

```
mounted /dev/sdb, volumeID is 561479FB832DCC526B1D7EDCD06B83E1
```

The following message appears when removing a volume:

```
removed /dev/sdb, volumeID was 561479FB832DCC526B1D7EDCD06B83E1
```

Note

These messages appear at the **announcement** level. Additional debug level messages appear in the syslog.

Node Status Reporting

Troubleshoot node errors and announcements by viewing the reporting sections in the Node Status page. Access these sections at the bottom of the Node Status page. The information in each section can be helpful when working with Swarm Support to resolve an issue.

Node Info section

The Node Info status section contains general information about the hardware installed on the node, as well as time server information and current uptime. Use this status information to determine if a node requires additional hardware resources.

The Swarm Admin Console generates an alert indicating the node may require additional RAM to maintain cluster performance if the **Index Utilization** and **Buffer Utilization** values rise to 80% or more. The node may not be communicating properly with an NTP server if the **Time** value does not match the same value in the remaining cluster nodes.

Node Info	
Cluster Name	swarm1.tx.caringo.com
Cluster Multicast IP	225.22.22.33
Version	7.0.0
Revision	69526.g13b1a50
Uptime	5 hrs, 26 mins, 27 secs
Time	04/30/14 21:20:09 GMT
Index Utilization	0%
Index Slots Available	13.42 million
Index Memory Reserved	494.8 MB
Reserve Memory	30.00 MB
IO Buffer Memory	299.9 MB
Accounted Memory Highwater	618352 bytes
Accounted Memory In Use	66716 bytes
Accounted Memory Utilization	0%

Additional Node Info reports

Scroll to the bottom of the **Node Info** section to access these links to additional reports:

- **SNMP Repository** (the SNMP repository dump)
- **Object Counts** (the Python classes in use)
- **Uncollectable Garbage**
- **HTML Templates**
- **Loggers...** (the settings window for changing the logging levels)
- **Dmesg dump** (the last 1000 messages logged by the Linux kernel reading buffer, for diagnosing a Swarm issue when a system panic or error occurs)
- **Hwinfo dump** (the Linux hardware detection tool output)

Node Configuration section

The Node Configuration status section contains the cluster and network configuration settings assigned to the node. Use this status information to quickly verify the system configuration without using SNMP commands.

Node Configuration	
bidding.optimizeMulticast	False
bidding.readBidOverride	0
bidding.relocationThreshold	5
bidding.writeBidOverride	0
cache.expirationTime	600
cache.maxCacheableSize	1048576
cache.percentage	10
cache.realmStaleTimeout	600
chassis.processes	1
cip.group	225.22.22.33
cip.histogramMinSamples	200
cip.histogramRateBuckets	5
cip.queryTimeout	0.03
cip.readBufferSize	1048576
cip.ttl	1
cluster.enforceTenancy	0
cluster.name	swarm1.tx.caringo.com
cluster.proxyIPAddress	
cluster.proxyPort	80
cluster.scspHoldDomain	scsp_hold
cluster.scspHoldMaxItems	1000
cluster.settingsUUID	a31bd4ddc395bf1015154b309003a340
cluster.settingsUuid	a31bd4ddc395bf1015154b309003a340

Node Operations section

The Node Operations status section describes the state of the node. A Swarm Support representative can use the information in this page to assist in determining if the node is communicating effectively with other nodes and resources in the cluster if a problem is encountered a problem in a storage cluster.

Some cluster features (such as the **Capacity** column value in the Swarm Admin Console) do not update until the HP cycles are completed separately on each node. The **HP Cycle time** parameter increases exponentially as the number of objects increase on the node. The node may not be servicing new requests if the **SCSP Last read bid** and **SCSP Last write bid** parameters are high.

Node Operations	
CAStor revision	69526.g13b1a50
CAStor version	7.0.0
HP Exam queue count	0
HP Replication queue count	0
HP last cycle, EC siblings: Degraded	0
HP last cycle, EC siblings: Maintenance impacted	0
HP last cycle, EC siblings: Missing	0
HP last cycle, EC siblings: Multicasts	0
HP last cycle, EC siblings: Relocated	0
HP last cycle, EC siblings: Total	0
HP last cycle, whole reps: Deleted stream	0
HP last cycle, whole reps: Maintenance impacted	0
HP last cycle, whole reps: Multicasts first exam	0
HP last cycle, whole reps: Multicasts frequency setting	9
HP last cycle, whole reps: Multicasts old version	0
HP last cycle, whole reps: Multicasts poor distribution	0
HP last cycle, whole reps: Multicasts requested exam	0
HP last cycle, whole reps: Multicasts stale hints	0
HP last cycle, whole reps: Multicasts too few answers	0
HP last cycle, whole reps: Multicasts too few hints	8

Hardware Status section

The Hardware Status section contains status and operational reporting (if available) for various hardware components installed on the node. Use this status information to retrieve node system data, such as the serial number and BIOS version.

Hardware status reporting is dependent on hardware supporting and populating [IPMI](#) sensors, [SMART](#) status, and, in some cases, manufacturer-specific components such as [SAS](#). Not all status fields are populated depending on the hardware. The hardware status values are independently scanned and populated for each node, allowing variations in supported utilities on a node-by-node basis.

Hardware Status	
Errors during probe	0
Temperature (Celsius)	0
Time of Last Probe	04/30/14 21:26:38
1398873212.01	
License status.	
isCFS	no
isCSN	no
isDX	no
isSCN	no
isSN	no
Generic system configuration	
BiosDate	11/04/2009
BiosVersion	F6
Manufacturer	Gigabyte Technology Co., Ltd.
ProductName	G41M-ES2L
SerialNumber	

Additional Hardware Status reports

Scroll to the bottom of the **Hardware Status** section to access these links to additional reports:

- **Test Network** - Pings all nodes in the cluster to verify all nodes can communicate with each other using TCP/IP and UDP (see details below).
- **Test Volumes** - Pings the volumes on the local hard drives and provides a response time (in milliseconds).
- **Dmesg Dump** - Displays the last 1000 messages logged by the Linux kernel reading buffer. These messages can help troubleshoot and diagnose a Swarm issue when a system panic or error occurs.
- **Hwinfo dump** (the Linux hardware detection tool output)
- **Send Health Report** (script that sends the hardware health report to the configured destination)

Test Network

Test Network performs two sets of tests:

- First, it sends 100 UDP multicasts to the cluster and computes the results:
- Which nodes responded
- How many responses returned
- How long the responses took, on average

- Next, it fetches the status page (port 80) via TCP for all responding nodes (once for each node). It tracks the total time for each of those round trips.

The data in the **Network Test Results** window allows comparing the responding nodes with the list of expected nodes in the cluster. Evaluate UDP packet loss and TCP connectivity within the cluster.

**Important**

A network issue may exist in the cluster if one or more nodes do not appear in the display.

Using Cluster Settings - Legacy Admin Console

Deprecated

The [Legacy Admin Console \(port 90\)](#) is still available but has been replaced by the [Swarm Storage UI](#). (v10.0)

- [Deprecated](#)
- [Caution](#)
- [Prior versions](#)
- [Changing the cluster name](#)
 - [Important](#)
- [Changing the multicast address](#)
 - [Important](#)
- [Logging setting](#)
- [Replication setting](#)
- [Suspend setting](#)
 - [Important](#)
- [Power setting](#)
 - [Best practice](#)
- [Cluster Domains setting](#)
- [Feeds setting](#)

The Cluster Settings dialog box appears, enabling runtime changes to several [configuration settings](#) in a cluster when clicking **Settings** in the Swarm Admin Console. The **Cluster Settings UUID** field displays the universal unique identifier (UUID) of the aliased object containing the cluster settings when saving settings the first time. All changes persist in the cluster across reboots.

Cluster Settings UUID:

Logging: Host:
 Port:
 Level:
 Audit Logging:

Replication: Multicast: %

Suspend: Volume Recovery

Power: Full Performance Mode
 Power Saving Mode
 Sleep after mins
 Wake after mins

Cluster Domains	Protection Setting
customdomain3-1446748571422.generic.main.caringo.com	Custom policy
customdomain3-1446749784366.generic.main.caringo.com	Custom policy
scsp_hold_550116c87b8f7cf54744c71c70a5c064	Only users in domain scsp_hold_550116c87b8f7cf54744c71c70a5c064/_administr...

Feed Name	Type	Default	Scope	Id	Delete	
Indexer	Search	<input checked="" type="radio"/>	global	0	<input type="checkbox"/>	<input type="button" value="Edit..."/>
Replicator1	Replication		global	1	<input type="checkbox"/>	<input type="button" value="Edit..."/>

Caution

Do not set cluster-wide persisted settings in the individual node configuration files because these values *must* be the identical for the entire cluster. Any values specified in a node configuration file are overwritten by the cluster settings in the Swarm Admin Console.

Prior versions

As of Swarm version 6.0, the cluster settings are automatically propagated to all nodes in the cluster. Adding the cluster settings UUID to the node or cluster configuration files are not required.

The first time changing a cluster-wide setting, the cluster settings UUID is automatically generated and propagated to all nodes in the cluster. Changing cluster-wide settings in the Swarm Admin Console by manually or programmatically adding a domain (see [Managing Domains](#)) or using `snmpset` (see [Using SNMP with Swarm](#)).

The `clusterSettingsUUID` [setting](#) value in the node or cluster configuration file is used when the node initially boots in version 6.0 if upgrading from an earlier Swarm version. After at least one node has booted to version 6.0, remove the `clusterSettingsUUID` setting from the node or cluster configuration file.

Changing the cluster name

Change the cluster name in the `cluster.name` [configuration setting](#) located in the node or cluster configuration file. Follow the guidelines below.

Important

Neither the cluster name nor the cluster multicast address can be changed at runtime.

- Set the `cluster.name` configuration setting value *before* changing any cluster settings. After the cluster settings UUID is set, do not change the value.
- Conflicts may occur in the settings UUID. Contact DataCore Support for instructions on resetting the cluster settings UUID if setting the `cluster.name` and changing it or having no `cluster.name` set and adding one after obtaining a cluster settings UUID, .
- The volume retains the original cluster settings and `cluster.name` setting value if moving a volume between clusters. To overwrite the original settings with the new cluster settings, the volume *must be* inserted into the new location while the remaining nodes are running. The settings from the originating cluster are detected first and become "master" in the new cluster if all volumes are mounted at the same time.

Changing the multicast address

The cluster nodes use a single multicast IP address to broadcast cluster-wide messages. To change the IP address, modify the `cip.group` [configuration setting](#) in the node or cluster configuration file.

Important

Neither the cluster name nor the cluster multicast address can be changed at runtime.

Logging setting

Update the logging **Host**, **Port**, and **Level** options to temporarily or permanently redirect the cluster logs to a different location or log level. This can be useful when troubleshooting issues in the cluster requiring a more granular report of cluster activity.

In addition, check the **Audit Logging** checkbox to send audit level events to the syslog, independent of the log level.

Replication setting

This option allows setting the frequency of multicast broadcasts within the cluster.

The higher the percentage entered in the **Multicast %** field, the more frequently the cluster communicates UUIDs to the cluster nodes and other services.

Suspend setting

Swarm automatically initiates recovery of replicas and erasure-coded segments known to be on a cluster volume no longer present in the cluster. Two recovery processes, failed volume recovery (FVR) and erasure coding recovery (ECR), are started for every volume detected as missing. Swarm announces both when it starts each process as well as when each completes, which may be relatively quick if there are no replicas or segments to recover.

The Suspend option allows to temporary suspension of both volume recovery processes in the cluster for situations where data is not actually at risk, but a network or power outages have taken one or more nodes (or an entire sub-cluster) offline. Suspending volume recovery by selecting the **Volume Recovery** check box prevents cluster activity churn and reduces the risk of capacity issues due to over-replication during an outage or upgrade.



Important

Do not suspend volume recovery indefinitely, as this hampers one of Swarm's primary data protection layers.

Power setting

This option allows selection of either a full performance mode or a power-saving mode.

- **Full Performance Mode** disables the power-saving settings. All nodes remain active (do not idle) and volumes become idle after at least six minutes of no I/O activity.
- **Power Saving Mode** enables the **Sleep After** and **Wake after** settings.
- **Sleep After** (the `power.sleepAfter` [setting](#)) sets the length of time without SCSP activity before health processing is paused and the node displays as Idle in the Swarm Admin Console.
- **Wake After** (the `power.wakeAfter` [setting](#)) sets the length of time a node remains idle before the health processor is reactivated.



Best practice

Use Full Performance Mode if the cluster is in constant use (24x7) or if uninterrupted feed restarts are critical for operations.

Cluster Domains setting

Cluster domains are secure domains existing entirely within a Swarm storage cluster. Similar to a storage facility containing multiple storage units, domains contain multiple buckets allowing storage of unstructured data objects in specific categories, such as documents, photos, and videos.

See [Managing Domains](#).

Feeds setting

Feeds is an object routing mechanism in the Swarm storage cluster using intermittent channel connections to distribute data to one or more targeted Elasticsearch servers or target clusters. The source cluster processes all UUIDs and names stored in the source cluster based on the feed configurations. This section lists the feeds configured for the source cluster.

See [Managing Feeds](#) for configuring replication and search feeds.

Viewing and Managing the Cluster - Legacy Admin Console



Deprecated

The [Legacy Admin Console \(port 90\)](#) is still available but has been replaced by the [Swarm Storage UI](#). (v10.0)

- [Deprecated](#)
- [Authenticating Cluster-wide Actions](#)
 - [Note](#)
- [Shutting Down or Restarting the Cluster](#)
 - [Note](#)
- [Finding Nodes in the Cluster](#)
 - [Listing of nodes](#)
 - [Prior versions](#)
 - [Finding nodes by IP](#)
 - [Finding nodes by Status](#)
 - [Note](#)
- [Percent Used Indicator](#)
- [Displaying Subcluster Information](#)
 - [Tip](#)

This section describes how to manage and maintain the cluster using the legacy Admin Console.

The **Cluster Status** page appears when logging in to the legacy Admin Console, providing a comprehensive view of the cluster as a whole and enabling cluster-wide actions, such as restarting and shutting down the cluster and modifying the cluster settings.

Authenticating Cluster-wide Actions

Shutting down and restarting the cluster are cluster-wide actions that require authentication. Authentication is also required when changing the cluster settings to:

- Manage domains
- Modify the logging host configuration
- Change the replication multicast value
- Suspend or resume volume recovery
- Set the power-saving mode

To commit the cluster-wide actions:

1. Open the node and/or cluster configuration file with the appropriate user credentials.
2. Modify the **security.administrators** [parameter](#).

An **admin user** is predefined in the parameter that allows **Basic** authentication by default. The password is sent in clear text from the browser to the legacy Admin Console.

Take the following precautions for added security:

1. Change the admin password *immediately*.
2. Implement real user names.
3. Encrypt user passwords using **Digest** authentication.

See [Encrypting Swarm passwords](#).

Note

After 60 seconds in most browsers, the user name and password are no longer valid. Use caution when entering the name and password in Safari and Chrome browsers, where information may *not* time out after 60 seconds.

Shutting Down or Restarting the Cluster

To shut down or restart all nodes in the cluster,

1. Log in to the legacy Admin Console with admin credentials.
2. Click **Shutdown Cluster** or **Restart Cluster** in the console.
3. When prompted, verify the procedure.

Note

Allow several minutes for the nodes to shut down or restart.

Finding Nodes in the Cluster

Listing of nodes

The cluster node list provides a high-level view of the active nodes in the system. Click the maximize button next to each IP address in the Node IP column to view the storage volumes within each cluster node.

07/01/16 18:38:58 GMT Licensed to: Caringo, Inc.

Node IP: Search Status: View All Health Report | Feeds | Print | Settings | Shutdown cluster | Restart cluster

Cluster Name	Nodes	Status	Errors	Streams	Used	Available	Capacity	Licensed	% Used
jans 225.0.10.104	4	Ok	0	1.67 million	252.7 GB	3.387 TB	3.771 TB	128.0 TB	10%

Node IP	Volumes	Status	Errors	Streams	Used	Available	Capacity	Licensed	Uptime	Version
172.30.11.33	2	Ok	0	418229	62.58 GB	848.1 GB	942.7 GB		1 day, 2 hrs	8.2.0
172.30.11.37	2	Ok	0	419382	63.16 GB	845.6 GB	942.7 GB		1 day, 2 hrs	8.2.0
172.30.11.41	2	Ok	0	416151	64.08 GB	845.8 GB	942.7 GB		1 day, 2 hrs	8.2.0
172.30.11.45	2	Ok	0	419503	62.83 GB	847.1 GB	942.7 GB		1 day, 2 hrs	8.2.0

© Copyright 2008-2014 Caringo, Inc. All rights reserved.

The status information is transmitted periodically to the legacy Admin Console, requiring up to two minutes before the node data in the Cluster Status window is updated. Because of the status propagation delay, the data for each node may vary in comparison. Remain connected to the same node to avoid

confusion for best results.

The volume labels next to each Swarm node are listed in arbitrary order. While the legacy Admin Console labels do not correspond to physical drive slots in node chassis, the volume names match the physical drives in the machine chassis. Each subcluster name must be expanded to display the corresponding volume information if the cluster is configured to use subclusters.



Prior versions

Nodes running legacy software versions (up to version 3.0) in a mixed version configuration may not display all data in the legacy Admin Console, such as object counts.

Search for nodes by IP address and by status if the cluster is large.

Finding nodes by IP

Search for a node using the Node IP search field in the legacy Admin Console for large clusters with multiple nodes.

Enter the node IP address in the field and click **Search** to locate the targeted node.

Finding nodes by Status

Select a **Status** from the drop-down menu to display nodes or volumes with a specific status.



Note

The overall cluster status is a roll-up of the statuses from cluster nodes.

Statuses include:

- **OK:** The node is working and there are no errors.
- **Alert, Warning:** The node or volume has experienced one or more errors. Click the IP Address link to drill down to the node and view the related error.
- **Initializing:** The short state after a node boots when it is reading cluster persisted settings and is not quite ready to accept requests.
- **Maintenance:** The node has been shut down or rebooted by an administrator from either SNMP or the legacy Admin Console and should not be considered missing for recovery purposes. By default a node can be in a *Maintenance* state for 3 hours before it transitions to *Offline* and the cluster starts recovery of its content. *Maintenance* mode is *not* initialized when the power is manually cycled on the node outside of Swarm (either physically on the hardware or via a remote shutdown mechanism like iDRAC) or if there is a disk error; in both these instances recovery processes are started for the node unless recovery is suspended.
- **Mounting:** The node is mounting one or more volumes, including formatting the disk if it is new and reading all objects on the volume into the RAM index for faster access.
- **Offline:** The node or volume was previously but is no longer present in the cluster.
- **Retiring:** The node or volume is in the process of retiring, verifying all objects are fully protected elsewhere in the cluster and then removing them locally.
- **Retired:** The node or volume has completed the retiring process and may be removed from the cluster.
- **Idle:** The nodes or volumes are in power-saving mode due during a period of configurable inactivity. (See [Configuring Power Management.](#))

Only matching results appear on the console when a value in the drop-down menu is selected. Select **View All** to redisplay all nodes in the cluster when finished looking at the searched node(s).

Percent Used Indicator

The **% Used** indicator provides a helpful computation of cluster availability and licensed and total physical space for monitoring purposes. Space used is calculated against the *lesser* of the total physical space or the licensed space.

For example, in a cluster with 4 TB of physical space but only 2 TB of licensed space where 1.5 TB of space is used, the console reports **75% Space Used**.

The indicators include color highlighting, as described below.

Logical Threshold	Color *	Description	Default Threshold Value
OK	Green	Used space is less than the console.spaceWarnLevel configurable threshold.	At or above 75%
Warning	Yellow	Used space is less than the console.spaceErrorLevel and more than the spaceWarnLevel configurable thresholds.	Above 75% but at or below 90%
Error	Red	Used space is greater than or equal to the console.spaceErrorLevel configurable threshold.	Above 90%

* These default colors can be modified using custom style sheets.

Displaying Subcluster Information

The Node List is grouped first by subcluster name then by node IP address if the cluster contains subclusters. The subcluster name is an IP address if no subcluster name is specified in the node or cluster configuration file. The first row of each subcluster includes a roll up of the status for the nodes in the subcluster.

Example of two subclusters expanded to show member nodes:

Cluster Name: 225.222.222.222
Nodes: 136
Status: OK
Streams: 0
Used: 157.57 million
Available: 2.039 TB
Capacity: 36.72 TB
Licensed: 128.0 TB
% Used: 75%

Subcluster	Nodes	Status	Errors	Streams	Used	Available	Capacity	Licensed	Uptime	Version
subA	44	OK	0	59.52 million	639.6 GB	12.39 TB	13.11 TB			7.0.0
subB	48	OK	0	54.97 million	717.7 GB	12.74 TB	13.57 TB			7.0.0
subC	44	OK	0	52.08 million	682.0 GB	11.58 TB	12.36 TB			7.0.0
172.30.12.62		OK	0	269522	2.789 GB	309.2 GB	312.0 GB		5 hrs, 21 mins	7.0.0
172.30.12.63		OK	0	297873	2.924 GB	309.0 GB	312.0 GB		5 hrs, 21 mins	7.0.0
172.30.12.64		OK	0	286092	2.968 GB	309.0 GB	312.0 GB		5 hrs, 21 mins	7.0.0

The status information is transmitted periodically to the legacy Admin Console, requiring up to two minutes before the node data in the Cluster Status page is updated. The data for each node may vary in comparison because of the status propagation delay.

Tip

Remain connected to the same node to avoid confusion for best results.

UI Essentials - Legacy Admin Console

Deprecated

The Legacy Admin Console (port 90) is still available but has been replaced by the [Swarm Storage UI](#). (v10.0)

- [Accessing the UI](#)
- [Navigating the UI](#)
- [Branding the Admin Console](#)

Accessing the UI

To connect to the legacy Admin Console, enter one of the following URLs in a browser's address or location field:

CSN Platform Server

`http://{CSN.external.IP}:8090/services/storage`

No Platform Server

`http://{cluster}:90`

For example, if the Swarm node IP address is 10.20.30.101, enter:

`http://10.20.30.101:90`

i CLUSTER or <cluster> in a URL stands for <host>[:<port>], where host is a fully qualified domain name or IP address, plus a port number if other than 80. If the Host header does not match the domain name, override it with the domain= argument.

Navigating the UI

The legacy Admin Console displays two separate views of the cluster: cluster-level and node-level status.

Cluster Status Page

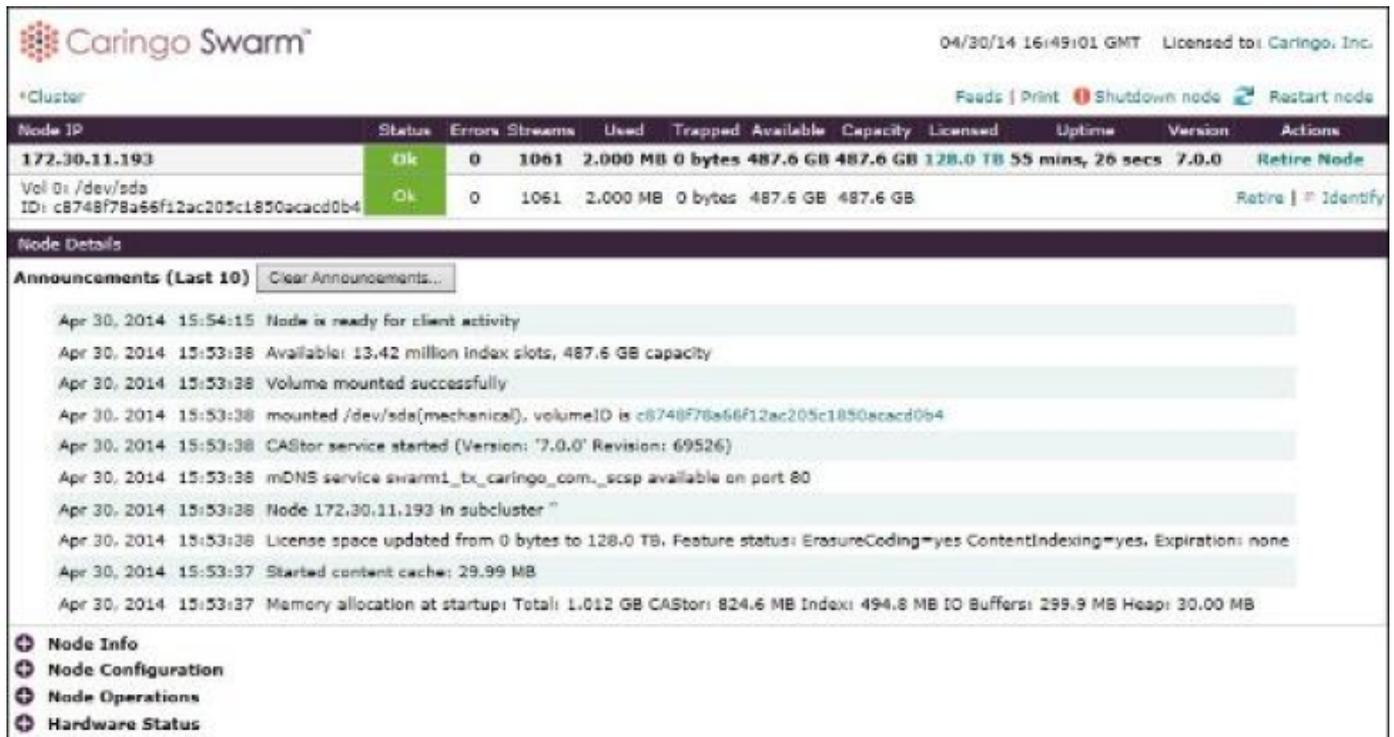
The Cluster Status page appears when initially logging in to the legacy Admin Console.

Cluster Name	Nodes	Status	Errors	Streams	Used	Available	Capacity	Licensed	% Used	
swarm1.tx.caringo.com 225.22.22.33	3	Ok	0	3174	6.000 MB	1.462 TB	1.463 TB	128.0 TB	0%	
Node IP	Volumes	Status	Errors	Streams	Used	Available	Capacity	Licensed	Uptime	Version
172.30.11.193	1	Ok	0	1061	2.000 MB	487.6 GB	487.6 GB		40 mins, 4 secs	7.0.0
172.30.11.194	1	Ok	0	1062	2.000 MB	487.6 GB	487.6 GB		40 mins, 4 secs	7.0.0
172.30.11.195	1	Ok	0	1051	2.000 MB	486.8 GB	487.6 GB		40 mins, 3 secs	7.0.0

The Cluster Status page allow viewing cluster-wide information such as number of nodes, cluster status, number of errors, number of streams (data components that comprise objects), and capacity data. If the cluster is configured with subclusters, you must expand a subcluster node name to display the associated IP addresses and then click an IP address to display node information.

Node Status Page

When clicking an IP address in the Node IP column, the Node Status page appears for the selected node, as shown below.



Node IP	Status	Errors	Streams	Used	Trapped	Available	Capacity	Licensed	Uptime	Version	Actions
172.30.11.193	Ok	0	1061	2,000 MB	0 bytes	487.6 GB	487.6 GB	128.0 TB	55 mins, 26 secs	7.0.0	Retire Node
Vol D: /dev/sda ID: c8748f78a66f12ac205c1850acacd0b4	Ok	0	1061	2,000 MB	0 bytes	487.6 GB	487.6 GB				Retire Identify

Node Details

Announcements (Last 10)

- Apr 30, 2014 15:54:15 Node is ready for client activity
- Apr 30, 2014 15:53:38 Available: 13.42 million index slots, 487.6 GB capacity
- Apr 30, 2014 15:53:38 Volume mounted successfully
- Apr 30, 2014 15:53:38 mounted /dev/sda(mechanical), volumeID is c8748f78a66f12ac205c1850acacd0b4
- Apr 30, 2014 15:53:38 CASTor service started (Version: '7.0.0' Revision: 69526)
- Apr 30, 2014 15:53:38 mDNS service swarm1_tx_caringo_com_sscp available on port 80
- Apr 30, 2014 15:53:38 Node 172.30.11.193 in subcluster "
- Apr 30, 2014 15:53:38 License space updated from 0 bytes to 128.0 TB. Feature status: ErasureCoding=yes ContentIndexing=yes, Expiration: none
- Apr 30, 2014 15:53:37 Started content cache: 29.99 MB
- Apr 30, 2014 15:53:37 Memory allocation at startup: Total: 1.012 GB CASTor: 824.6 MB Index: 494.8 MB IO Buffers: 299.9 MB Heap: 30.00 MB

- Node Info
- Node Configuration
- Node Operations
- Hardware Status

The Node Status page allows viewing information specific to your cluster node, such as hardware status, health processor status, uptime, and Swarm software version.

See [Managing Chassis and Drives](#).

The Node Status page also includes the following sections:

- **Node Info.** Provides general information about the hardware installed on the node, as well as time server information and current uptime.
- **Node Configuration.** Provides the cluster and network configuration settings assigned to the node. Use this status information to verify your system configuration quickly, without using SNMP commands.
- **Node Operations.** Describes the state of the node. If you encounter a problem in your storage cluster, a Swarm Support representative can use the information in this page to help determine if the node is communicating effectively with other nodes and resources in the cluster.
- **Hardware Status.** Provides status and operational reporting (if available) for various hardware components installed on the node. Use this status information to retrieve node system data, such as the serial number and BIOS version.

Printing the legacy Admin Console

Click **Print** at the top of the page to display a printer-friendly version of the legacy Admin Console.

To include a portion of the console in an email, copy the text and paste it to an HTML-formatted email.

If you are not using HTML-formatted email, you can paste a portion of the printed formatted page into an application (such as Microsoft® Excel® or Word®) and then copy it to another location.

Tip

Print the legacy Admin Console in landscape mode to verify the image does not extend beyond the right margin if using Mozilla Firefox®.

Viewing License Information

Your cluster license appears when you click the **Licensed to** link at the top of the legacy Admin Console. This window displays your registration status, contact information, and the configuration settings in the license file.

License Registration	
Status	
Registration:	Registered
Serial Number:	20120703171500-2231
Expiration:	None
Licensed To	
Company:	Caringo, Inc.
Address:	6801 N. Capital of Texas Hwy, Suite 2-200
City:	Austin
State:	Texas
Zip:	78731
Country:	USA
Configuration	
Cluster Description:	Caringo Internal Testing Clusters
Licensed Capacity:	128.0 TB
Minimum Replicas:	1
Erasure Coding:	True
Content Indexing:	True

If your cluster license is invalid, **Unregistered** appears in the **Company** field and as a watermark in the legacy Admin Console. Contact your Swarm representative to purchase a valid license.

Branding the Admin Console

You can override the baseline style sheets in the legacy Admin Console using a centralized configuration.

The legacy Admin Console ships with a set of default styles. These styles are persisted in these files:

- **console.css.** Provides a set of baseline styles.
- **console_print.css.** Provides a small set of overrides for the printed page.
- **console_print_preview.css.** Provides the styles for the on-screen print view.

These files are loaded at boot time from a USB or centralized configuration `caringo.console` directory. If the files are not available in this location, Swarm uses internal file versions.

The styles defined in the standard files can be overwritten with custom styles on a style-by-style basis. Any styles not overwritten revert to the baseline styles provided in the default style sheets.

Tip

To make customization easier, use a centralized configuration web server.

If a centralized configuration server is not available for your cluster, you can update the console styles by modifying the default styles in **console.css**, **console_print.css**, and **console_print_preview.css** on the USB flash drive for each node.



Best practice

Before you modify the default file, create a backup of the file in case you need to revert to the default styles.

Overriding styles on a centralized server

To override the baseline styles using a centralized configuration server:

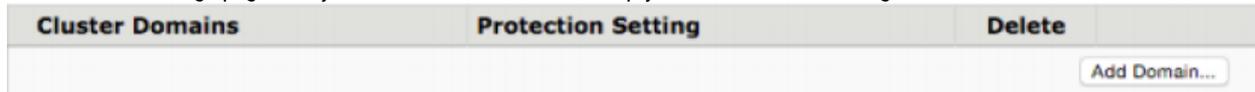
1. Create new style sheet files, clearly named to distinguish them from the originals. The new style sheets define one or more styles from `console.css`, `console_print.css`, or `console_print_preview.css` to override. At minimum, paste specific styles from the default style sheets into your new one and then change those style definitions.
2. Install your new styles and place them on any web server that the storage cluster can access.
3. Configure the cluster to reference your new styles with the `consoleStyleURL` and `consoleReportStyleURL` node [configuration parameters](#). When you override the styles, the `consoleReportStyleURL` parameter is used for both the `console_print.css` and `console_print_preview.css` style sheets.

Managing Domains - Legacy Admin Console

Gateway
Skip this section if your client applications are accessing your cluster via the [Content Gateway](#).

To add, edit, or delete a domain from the Swarm Admin Console:

1. Open the Swarm Admin Console, and click **Settings**.
2. In the Cluster Settings page, verify the Cluster Domains box is empty if no domains are configured.



3. To add a domain, click **Add Domain**.
To edit a domain, click **Edit** next to its name.
To delete a domain, select the check box next to its name, click **Delete**, and click **Submit**.

Note
If you delete a domain that contains buckets without including the recursive query argument, the buckets and any objects they contain are not deleted, but they are inaccessible. To work around this issue, see [Restoring Domains and Buckets](#).

4. Enter or edit the following information:

Option	Description
Domain Name field	<p>Enter a fully qualified IANA-compliant name to identify this domain (for example, cluster.example.com). The domain name must be unique among all clusters you manage.</p> <p>If you did not configure a domain name that matches your cluster's name, the cluster name displays in this field. Creating a domain with the same name as the cluster sets up a <i>default cluster domain</i>.</p> <p>See the Naming Rules. To rename an existing domain, use the Swarm Admin Console.</p>
Protection Setting	<p>Determines what users are authorized to POST to the domain (such as buckets and unnamed objects). Click one of the following:</p> <ul style="list-style-type: none"> • All Users. No authentication required. Any user can create buckets or unnamed objects in the domain without authentication. • Only users in this domain. Enables users in the user list associated with this domain to POST to the domain. • Only users in domain. Enables users in the user list associated with the specified domain to POST to this domain.

Domain Managers	Manages user lists in the domain. For help creating user lists, contact your Support representative. To add a new administrator, click Add Domain Manager . To edit an existing manager, click Edit next to the administrator's manager.
<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> <p>Important</p> <p>Domain manager names consist of ASCII characters only and <i>cannot</i> include a colon character (:). See the Naming Rules.</p> </div> <div style="border: 1px solid #ccc; padding: 5px;"> <p>Tip</p> <p>If Custom Policy displays for Protection Setting, a domain administrator has manually altered the protection settings. Set the protection setting back to the default setting if attempting to troubleshoot an issue with users being able to access objects in a domain.</p> </div>	

5. If you are adding or editing a domain manager, enter or edit the following information:
 - **User ID.** Enter a name to identify the domain manager. Domain manager names consist of ASCII characters only and *cannot* include a colon character (:).
 - **Password.** Enter a password for the domain manager.
6. Click **Submit**.
7. If prompted, enter an administrative user name and password.

This is how a domain appears in the Cluster Settings page:

Cluster Domains	Protection Setting	Delete	
swarm1.tx.caringo.com	Only users in domain swarm1.tx.caringo.com	<input type="checkbox"/>	Edit...
			Add Domain...

Note

The following error indicates that the previous domain name has not expired from the content cache: `ERROR! realm name 'domain-name/_administrators' already exists.`

This error is displayed if a cluster administrator renamed **cluster.example.com** to **domain.example.com** and attempted to create **cluster.example.com** before the name has expired from the content cache. . Wait several minutes and attempt again.

Viewing and Editing Feeds - Legacy Admin Console

Deprecated

The [Legacy Admin Console \(port 90\)](#) is still available but has been replaced by the [Swarm Storage UI](#). (v10.0)

- [Viewing Cluster Feeds](#)
- [Adding a feed](#)
- [Feed Actions](#)

There are three types of feeds be created in Swarm:

1. **Replication**
2. **Search**
3. **S3 Backup** – see [S3 Backup Feeds](#). (v11.0)

A **Replication Feed** performs replication continuously and adaptively:

- Replicates all objects in the source cluster or domain.
- Performs continuous replication to keep up with source cluster intake, within available connectivity and bandwidth to the target cluster.
- Verifies object replication to the targeted cluster as soon as possible.
- Uses an intermittent connection (such as HTTP) to move content from the source to the target cluster.

A **Search Feed** is an object-routing mechanism in the storage cluster using intermittent channel connections to distribute data to Elasticsearch for object metadata searching. The source cluster processes all UUIDs and names stored in the source cluster based on the feed configurations in the Swarm Admin Console **Settings** page. As objects are added to the cluster, Swarm adds the UUIDs and names to the assigned feed queue and notifies the target Elasticsearch server feed data is available.

Note

Multiple search feeds can exist to populate different ES clusters, but one can be designated as the primary, for searching from Swarm.

Viewing Cluster Feeds

The Cluster Feeds page appears when clicking **Feeds** in the Swarm Admin Console. Feeds are an object routing mechanism in the storage cluster using intermittent channel connections to distribute data to one or more targeted storage clusters.

The following can be sent using Feeds:

- Metadata content from the source cluster to the search servers (Search Feed)
- Objects from the source cluster or a particular domain to a targeted cluster (Replication Feed)

Column	Description
Feed Name / Feed ID	The name of the feed and the corresponding number uniquely identifying the feed.

Estimated Completion Time	The estimated time required to process and clear the current backlog of replication feeds based on the last 60 minutes. This estimate assumes conditions do not change.
Percent Completed	The dynamic report of progress to completion, as a percentage.
Avg. Objects / Min	The average number of objects processed per minute based on the processing rate during the last 60 minutes.
Nodes Reporting	The number of cluster nodes that sent feed reports to be aggregated.

Each feed provides color status indicators showing the current state. The following table describes the status colors and corresponding states for search and replication feeds.

Color	State	Description
Green	OK	The feed replicated successfully to the targeted search server or cluster node.
Gray	Paused	The feed is suspended because of an ongoing fast volume recovery (FVR) issue on the node.
Orange	Blocked	The destination node or cluster is offline or not accepting feeds. <ul style="list-style-type: none"> For Search feeds, the search service may be offline. For Replication feeds, the remote cluster may be offline.
Red	Config Error	The feed is misconfigured.

Viewing a replication feed

The Replication Feed window provides the statistics for a replication feed sent to a targeted Replication cluster. Depending on the configuration, a typical configuration can have more than one replication feed.

Swarm logs a critical message and updates the status color and state in the window for quick identification a problem when an issue occurs for a particular Replication feed.

The following data appears when expanding the Replication Feed:

Column	Description and States												
<table border="1"> <thead> <tr> <th>Feed Name</th> <th>Feed Id</th> <th>Est. Completion Time</th> <th>Percent Completed</th> <th>Avg. Obj./Min./Node</th> <th>Nodes Reporting</th> </tr> </thead> <tbody> <tr> <td>Disaster Recovery</td> <td>0</td> <td>18 secs</td> <td>90.909%</td> <td>19.9</td> <td>2 of 2</td> </tr> </tbody> </table>	Feed Name	Feed Id	Est. Completion Time	Percent Completed	Avg. Obj./Min./Node	Nodes Reporting	Disaster Recovery	0	18 secs	90.909%	19.9	2 of 2	<p>The current feed processing state to the Replication cluster. The state can be:</p> <ul style="list-style-type: none"> OK. Operating normally. Paused. Temporarily paused due to volume recovery. Blocked. Processing blocked due to a transient condition. The Node Plugin State indicates Blocked as well. (See below.)
Feed Name	Feed Id	Est. Completion Time	Percent Completed	Avg. Obj./Min./Node	Nodes Reporting								
Disaster Recovery	0	18 secs	90.909%	19.9	2 of 2								
<table border="1"> <thead> <tr> <th>Node Plugin State</th> <th>State</th> </tr> </thead> <tbody> <tr> <td>ok</td> <td>gathering</td> </tr> </tbody> </table>	Node Plugin State	State	ok	gathering	<p>The HTTP transaction state to the Replication cluster. The state can be:</p> <ul style="list-style-type: none"> Idle. No pending work or replication at this time. Gathering. Gathering replication entries for a future request. Replicating. Processing ongoing requests to the destination cluster. Blocked. Waiting for a transient error to clear. (See below.) Permanent Error. Permanently stuck due to configuration errors. 								
Node Plugin State	State												
ok	gathering												

Nodes Reporting	<p>The IP addresses of each reporting node contributing to the aggregate report.</p> <p>Nodes appear together as a group if all target cluster nodes report successfully. Nodes appear in separate rows with corresponding feed state if one or more nodes does not report successfully.</p> <p>Each IP address hyperlinks to the SNMP Repository Dump page for the reporting node, which provides node-specific detail for each feed.</p>
Aggregated Current Object Processing	<p>Provides the aggregated statistics for new or updated objects. These statistics include:</p> <ul style="list-style-type: none"> • Pending Evaluation. Objects evaluated before assignment to the replication feed. • Unprocessed. Objects queued for processing by the target cluster. • Successes. Objects replicated by the target cluster. • Retrying. Objects rejected by the target cluster due to a server or network problem. These objects continue to be transmitted until they are accepted by the target cluster.
Aggregated Versioned Object Processing	<p>Provides the aggregated statistics for versioned objects:</p> <ul style="list-style-type: none"> • Pending Evaluation. Versions evaluated before assignment to the replication feed. • Unprocessed. Versions queued for processing by the target cluster. • Successes. Versions processed by the target cluster. • Retrying. Versions rejected by the target cluster, due to a server or network problem. These objects continue to be transmitted until they are accepted.
Aggregated Delete Event Processing	<p>Provides the aggregated statistics for deleted objects. These statistics include:</p> <ul style="list-style-type: none"> • Pending Evaluation. Object deletes received but not evaluated by the target cluster. • Unprocessed. Object deletes queued for processing by the target cluster. • Successes. Object deletes replicated by the target cluster. • Retrying. Object deletes not processed by the target cluster due to a server or network problem. The object deletes continue to be transmitted until they are processed by the target cluster.

Viewing a search feed

The Search Feed window provides statistics for a search feed sent to a targeted Elasticsearch cluster. A typical configuration has one search feed.

Swarm logs a critical message and updates the status color and state in the window for quick identification a problem when an issue occurs for a particular search feed.

Note

Elasticsearch proactively monitors the disk space on the nodes. Elasticsearch suspends itself on nodes as soon as space falls below 5% and automatically unblocks itself when space becomes available, without requiring Swarm to restart.

The following status summaries appear when expanding the Search Feed:

Column	Description and States
--------	------------------------

Feed Name	Feed Id	Est. Completion Time	Percent Completed	Avg. Objs./Min./Node	Nodes Reporting
elastic1	1	0	100%	12.7	4 of 4

Node Feed State	Node Plugin State	Nodes Reporting
ok	idle (ES:yellow)	4 (100%)
ok	indexing (ES:yellow)	1 (25%)

Node Plugin State	Total in Process	Total Completed
Pending	0	1.18 million
Processing	0	1.18 million
Retrying	0	1.18 million
Total	0	1.18 million

Aggregated Current Object Processing	Aggregated Versioned Object Processing	Aggregated Delete Event Processing
Total in Process: 0	Total in Process: 5	Total in Process: 2
Pending: 0	Pending: 0	Pending: 0
Processing: 0	Processing: 5	Processing: 2
Retrying: 0	Retrying: 0	Retrying: 0
Total Completed: 1.18 million	Total Completed: 39850	Total Completed: 240605
Total : 1.18 million	Total : 58061	Total : 240607

Node Feed State	Description
OK	Operating normally.
Paused	Temporarily paused due to volume recovery.
Blocked	Blocked due to a transient condition. The Node Plugin State indicates Blocked as well. (See below.)
Idle	No pending work or replication at this time.
Indexing	Processing ongoing requests to the search server.
Blocked	Waiting for a transient error to clear. (See below.)
Permanent Error	Permanently stuck due to configuration errors.

Nodes Reporting	Description
The number of nodes and corresponding IP address contributing to the aggregate report.	
Nodes appear together as a group if the search nodes report successfully. Nodes appear in separate rows with corresponding feed state if one or more nodes does not report successfully.	
Each IP address hyperlinks to the SNMP Repository Dump page for the reporting node, which provides node-specific detail for each feed.	

Aggregated Current Object Processing	Description
Provides the aggregated statistics for new or updated objects:	
<ul style="list-style-type: none"> Pending Evaluation. Objects evaluated before assignment to the search feed. Unprocessed. Objects queued for processing by the search service. Successes. Objects processed by the search service. Retrying. Objects rejected by the search service, due to a server or network problem. These objects continue to be transmitted until they are accepted. 	

Aggregated Versioned Object Processing	Description
Provides the aggregated statistics for versioned objects :	
<ul style="list-style-type: none"> Pending Evaluation. Versions evaluated before assignment to the search feed. Unprocessed. Versions queued for processing by the search service. Successes. Versions processed by the search service. Retrying. Versions rejected by the search service, due to a server or network problem. These objects continue to be transmitted until they are accepted. 	

Aggregated Delete Event Processing	Description
Provides the aggregated statistics for deleted objects:	
<ul style="list-style-type: none"> Pending Evaluation. Object deletes not belonging to the search feed. Unprocessed. Object deletes queued for processing by the search service. Successes. Object deletes processed by the search service. Retrying. Object deletes not processed by the search service, due to a server or network problem. The object deletes continue to be transmitted until they are processed. 	

Adding a feed

Implement Feeds through the **Cluster Settings** page.

Adding a replication feed

The **Add Replication Feed** dialog box allows entering the configuration settings for a replication feed to a target cluster.

Feed Name	The friendly name attached to this feed. This name appears in the Feeds row in the Settings page.
Replicate all objects	Enable to replicate <i>all</i> objects in the source cluster to the target cluster, regardless of domain.
Feed Name: <input type="text" value="DR Asia Pacific"/>	
Replicate all objects: <input type="checkbox"/>	The filtering options become available and must be populated with valid values if disabling this option.
Domains to replicate: <input type="text" value="acme.*,corp.*"/>	
Domains to exclude: <input type="text" value="*.secure"/>	Required. Specify one or more domains to include, by name (<code>hrfoo.example.com</code> , <code>itfoo.example.com</code>) or by wildcard (<code>. *foo</code>).
Propagate deletes: <input checked="" type="checkbox"/>	Important: To filter by exclusion, specify all (<code>. *</code>) or else no domains are allowed.
Replication mode: <input type="radio"/> Replicate via direct POST (recommended; supports Gateway) <input checked="" type="radio"/> Replicate via bidirectional GET	
Replication threads: <input type="text" value="6"/>	
Remote cluster proxy or cluster host(s): <input type="text" value="172.30.12.88"/>	Wildcards Pattern matching follows the Python regular expression (RE) syntax with the exception that the "{m,n}" repetitions qualifier may not be used.
Remote cluster proxy or cluster host(s) port: <input type="text" value="80"/>	
Remote cluster name: <input type="text" value="dr2ap1"/>	
Remote cluster administrative username: <input type="text" value="admin"/>	
Remote cluster administrative password: <input type="password" value="*****"/>	
Domains to exclude	Optional. Specify one or more domains to exclude from the set of domains to replicate, by name (<code>abc123.example.com</code> , <code>abc456.example.com</code>) or by wildcard (<code>abc.*</code>).
Replicate objects in no domain	Optional. Includes unnamed objects not tenanted in any domain.
Propagate deletes	Enable to keep object deletes synchronized with the source cluster. Disable to prevent objects from being deleted in the target cluster.
Replication mode	Specifies replication via direct POST (recommended) or bidirectional GET.
Replication threads	For best performance, choose direct POST replication, which can go through Gateway. GET replication is the legacy method, which may be needed for application compatibility or networking requirements. Switching modes does not require a feed restart. (v9.6)
Replication threads	<i>Replication via direct POST only.</i> The default replication speed (6 simultaneous threads) is best for same-sized clusters with minimal replication backlog. (v9.6)
Remote cluster proxy or cluster host(s)	To avoid overwhelming a smaller target cluster, reduce the threads. For faster replication against a backlog, increase the threads temporarily, but monitor bandwidth and cluster performance, as boosting the speed stresses both clusters.
Remote cluster proxy or cluster host(s)	The IP address of either: <ul style="list-style-type: none"> • One or more nodes in the target cluster. • A reverse proxy host that routes to the target cluster.
Remote cluster proxy or cluster host(s) port	To enter two or more node IP addresses, enter each address separated by a comma or spaces.
Remote cluster proxy or cluster host(s) port	Defaults to 80. Allows specifying a custom port for the remote cluster. (v9.6)
Remote cluster name	Defaults to 80. Allows specifying a custom port for the remote cluster. (v9.6)
Remote cluster name	The configuration setting for the target cluster (the <code>cluster.name</code> value in the <code>.cfg</code> file of the target cluster).

Remote cluster administrative user name	The administrative user name of the target cluster. Enter a value in this field <i>only if</i> the remote cluster user name is different from the source cluster name in the same realm.
Remote cluster Administrative password	The administrative password of the target cluster. Enter a value in these fields <i>only if</i> the remote cluster password is different from the password on the source cluster in the same realm.

To add a replication feed:

1. Open the Swarm Admin Console and click **Settings**.
2. In the Feed Name row, click **Add Replication**.
3. Enter the administrator name and password when prompted for authentication.
4. In the **Add Replication Feed** window, complete the fields as described above.
5. Click **Add**. The new feed appears in the **Feed Name** row in the **Cluster Settings** page and propagates to the targeted nodes in the cluster within 60 seconds.

Feed Name	Type	Default	Scope	Id	Delete
Searcher	Search	<input checked="" type="checkbox"/>	global	0	<input type="checkbox"/> Edit...
Replicator	Replication	<input type="checkbox"/>	global	1	<input type="checkbox"/> Edit...

6. Click **Update**. A **Success** dialog box appears.

7. Click **Close**.

Adding a search feed

The **Add Search Feed** dialog box allows entering the configuration settings for a search feed to the Elasticsearch server.

Feed Definition	
Feed Name:	<input type="text" value="Indexer1"/>
Search server(s):	<input type="text" value="indexer1, indexer2, indexer3"/>
Search server port:	<input type="text" value="9200"/>
Search full metadata:	<input checked="" type="checkbox"/>
Feed batch size:	<input type="text" value="1000"/>
Feed batch timeout:	<input type="text" value="1"/>
Pause feed:	<input type="checkbox"/>
Refresh feed:	<input type="checkbox"/>

The following table describes the data entry fields in the dialog box.

Field	Description
Feed name	The name attached to the feed.

Search server(s)	The IP addresses or server names resolvable by DNS. Separate each server name with a space if entering more than one server. DNS must be configured on both the source and target clusters.
Search server port	The default port for a host.
Search full metadata	Enabled - Swarm indexes all object metadata, including baseline and client metadata fields. Disabled - Swarm indexes the baseline metadata fields. See Metadata Field Matching for a list of baseline and custom fields.
Feed batch size	The maximum number of objects sent concurrently to be processed. The default is 100.
Feed batch timeout	The maximum amount of time (in seconds) before a batch is resent to be processed after a timeout. The default is 1.

To add a search feed

1. Open the Swarm Admin Console and click **Settings**.
2. In the **Feed Name** row, click **Add Search**.
3. Enter the administrator name and password when prompted for authentication.
4. In the **Add search feed** window, complete the fields as described above.
5. Click **Add**.
The new feed appears in the **Feed Name** row in the **Cluster Settings** page and propagates to the targeted nodes in the cluster within 60 seconds.
6. Click **Update**.
A **Success** dialog box appears.
7. Click **Close**.

Feed Actions

Deleting a replication feed

Source cluster resources are freed when deleting a feed. This process does not affect the objects previously pushed to the target cluster.

To delete a feed:

1. In the Swarm Admin Console click **Settings**.
2. In the **Cluster Settings** window, locate the **Feed Name** section.
3. Identify the feed to be deleted and select the corresponding **Delete** checkbox.
4. Click **Update**.
A **Success** dialog box appears.
5. Click **Close**.
The deleted feed is removed from the remaining cluster nodes within 60 seconds.

Editing a search feed

To edit a feed

1. Open the Swarm Admin Console, and click **Settings**.
2. In the **Cluster Settings** window, locate the **Feed Name** section.

3. Select the feed to edit and click **Edit**.
4. In the **Edit search feed** window, make any changes as needed to the appropriate fields.
5. Set the **Refresh feed** option based on the Elasticsearch server configuration:
 - Select **Refresh feed** to hydrate a new search server.
 - Deselect **Refresh feed** to update a running feed on a current search server.
6. Click **Update**.
Updates appear in the **Cluster Settings** window and propagate to the remaining cluster nodes within 60 seconds.

Pause feed

Pause the feed or unpause it by toggling the checkbox option when editing a feed.

Pause the search feed before stopping the Elasticsearch service in the search cluster when upgrading the Elasticsearch cluster.

See [Installing Elasticsearch](#).

Refresh feed

As objects are written or updated, metadata is sent to the search servers in near real-time (NRT). Any objects that cannot be processed immediately are retried each HP cycle until they succeed, at which point they are marked as complete and are not resent. *Refresh* the feed if a data loss failure occurs on the Elasticsearch servers and restore from backup is not possible, which verifies and rehydrates all metadata content.

Choose either to refresh all search data for the feed or to add the changes to the running feed, without data verification, when editing a feed.

- **Enabled** - Resends *all* object metadata to the Elasticsearch server. An index is created if an Elasticsearch index for the cluster does not exist. To recreate an existing index (such as for case-insensitive searching where case-sensitive was previously used), drop the existing index before refreshing the feed. Select **Refresh feed** to reverify the feed content on the new or moved server if a new Elasticsearch server is installed in the domain or are moving it to another domain.

 **Note**
Expect lower performance while the feed rebuilds the metadata index.

- **Disabled** - Adds changes to the running feed without resending previously processed objects. Deselect **Refresh feed** to add the changes to the running feed if updating current search service.

 **Best practice**

Swarm does not resend prior deletions of *unnamed objects* to the search service, and it resends prior deletions of *named objects* within 14 days of the deletion. Drop the search index before refreshing the feed if a large number of deleted objects need to be purge.

Deleting a search feed

Source cluster resources are freed when deleting a feed. This process does not affect the objects previously pushed to the search server.

 **Important**

Define or select another search feed and mark it as default if deleting the default search feed, to support Elasticsearch queries.

To delete a feed

1. Open the Swarm Admin Console, and click **Settings**.
2. In the **Cluster Settings** window, locate the **Feed Name** section.
3. Identify the feed to be delete and select the corresponding **Delete** checkbox.
4. In the **Security** dialog box, enter the administrator name and password and click **OK**.
5. Click **Update**. A **Success** dialog box appears.
6. Click **Close**. Changes propagate to the targeted nodes in the cluster within 60 seconds.

[Delete the search data](#) previously sent by the feed.

Troubleshooting Blocked Feeds

Troubleshoot the feed state to the Replication cluster by reviewing the Feeds Table data in the [SNMP Repository Dump](#) page for each reporting cluster node if the **Node Feed State** indicates **blocked**.

For more information about the [SNMP Repository Dump](#) tables, see the SNMP MIB Reference file included in the top level of the Swarm product distribution ZIP file.

To troubleshoot feeds

1. Log in to the Swarm Admin Console as an Administrator.
2. In the console, click **Feeds**.

Replication Feed					
Feed Name	Feed Id	Est. Completion Time	Percent Completed	Avg. Objects/Min.	Nodes Reporting
Replicator1	1	Unknown	0.000%	0.0	4 of 4
Node Feed State	Node Plugin State	Nodes Reporting			
blocked	blocked	4 (172.30.15.201, 172.30.15.207, 172.30.15.213, 172.30.15.215)			
Aggregated Object Event Processing		Aggregated Delete Event Processing			
Total In Process		5341	Total In Process		70
Pending Evaluation		4490	Pending Evaluation		70
Processing		851	Processing		0
Retrying		0	Retrying		0
Total Completed		0	Total Completed		0
Total		5341	Total		70

- In the Cluster Feeds window, the **orange** status highlights the blocked node feed and plugin states for the Feed.
- The **Nodes Reporting** box lists the IP addresses for any offline replication cluster nodes or nodes not accepting feeds.
- In the **Nodes Reporting** box, click a reporting node IP address.
The **Feed Table** in the [SNMP Repository Dump](#)

page appears for the selected node.

4. Review the **feedPluginState** status to identify the blockage.

Example:

feedNumDeletesQueued	0	0
feedNumExistsQueued	0	0
feedPluginState	idle (ES:green)	blocked: Destination cluster onyx1 reports invalid request: Castor-System-Cluster value must refer to a remote cluster on RETRIEVE request
feedQueueInService	0	1
feedQueueOldestEntry	0	0
feedState	ok	blocked
feedStreamsPerMinLastHour	80	0
feedType	Indexing	Replication

`feedPluginState` `blocked: Destination cluster onyx1 reports invalid request: Castor-`

5. Repeat to troubleshoot the remaining blocked IP addresses.

Identifying the Primary Search Feed

If you do not have access to the Swarm Storage UI, you can find Swarm's alias name for your primary search index using one of these methods:

Legacy Admin Console - View the SNMP Repository Dump to find out what search feeds are defined:

1. Open the dump via the legacy Admin Console: `http://{host_or_ip}:90/snmp_dump`
2. Click **Feed_Table** and look at the row for **feedDefinition**.
3. If you have more than one feed (column), the primary (default) feed is the one that does *not* have **'respondsToLists': False**:

Feed Table	
feedDefinition	<pre>{'destination': {'insertBatchTimeout': 1, 'host': 'ivory1 ivory2 ivory3', 'insertBatchSize': 100, 'indexAlias': 'megapius-1', 'fullMetadata': 1, 'port': 9200}, 'type': 'Indexing', 'name': 'Indexer', 'lastchanged': 'Mon, 05 Sep 2016 18:49:07 GMT'}</pre>
	<pre>{'destination': {'insertBatchTimeout': 1, 'host': 'ivory1 ivory2 ivory3', 'insertBatchSize': 100, 'indexAlias': 'megapius1', 'respondsToLists': False, 'fullMetadata': 1, 'port': 9200}, 'type': 'Indexing', 'name': 'Caringo Search - new', 'lastchanged': 'Thu, 27 Oct 2016 17:03:37 GMT'}</pre>

4. For the primary feed, note the value for **'IndexAlias'** and enter this in the NFS export definition.

curl - Query to find out what search feeds are defined:

```
curl http://{host_or_ip}:91/api/storage/feeds?pretty=true
```

If more than one is listed, query to find out which one is the primary (default) feed:

```
curl http://{host_or_ip}:91/api/storage/feeds/0?pretty=true | grep "isDefault|indexAlias"
curl http://{host_or_ip}:91/api/storage/feeds/1?pretty=true | grep "isDefault|indexAlias"
```

Elasticsearch for Swarm

Elasticsearch provides Swarm the capability for metadata searching and historical metrics, and it furnishes the data needed to populate the [Swarm Storage UI](#).

This section covers implementing and maintaining Elasticsearch (ES) with Swarm.

For the query arguments for listing and search operations on object metadata, see [Search Query Arguments](#).

Terms of Use

Swarm keeps the SCSP searching API stable and insulates applications from ES schema changes.



Warning

DataCore may modify, without notice, the schema of the information contained within Elasticsearch and of the mapping template. Such changes can affect an implementation if directly interfacing with Elasticsearch or change the template included with the Swarm version of the Elasticsearch software.

Guidelines - follow these guidelines if a need exists to customize schemas, templates, or queries using Elasticsearch with Swarm:

1. *Swarm does not support customized schemas.* Run a separate ES instance for that purpose if schema changes are needed for direct-to-ES projects.
2. Because Swarm upgrades may include schema changes, always plan to test and adjust any custom direct-to-ES queries used. Consider running a separate ES instance if an extensive need exists for direct-to-ES operations.
3. Use a test environment to verify an update works with direct-to-ES queries.

- [Snapshot and Restore Search Data](#)
- [Rebuilding a Search Feed](#)
- [Monitoring Elasticsearch](#)
- [Adding Nodes to an ES Cluster](#)
- [Resetting Elasticsearch](#)
- [Rolling Restart of Elasticsearch](#)
- [Uninstalling Elasticsearch](#)
- [Merging and Renaming ES Clusters](#)

Snapshot and Restore Search Data

- [Note](#)
- [Important](#)
- [Configuring the Plugin](#)
 - [Best practice](#)
- [Configure the S3 Repository](#)
- [Creating a Snapshot the S3 Repository](#)
- [Restoring from a Snapshot](#)

Note

This technique makes use of and requires the [Content Gateway](#) with S3 enabled.

Swarm builds and maintains your search data (*index*) through your [Search Feed](#), and it regenerates the search index should it ever be lost. You can trigger this regeneration at any time by running the **Refresh** command for your search feed in the Swarm Storage UI (or legacy Admin Console, port 90). A complete refresh (which verifies all data) takes a long time, during which your listings are unavailable.

A method exists to take a snapshot of the index data so it can be restored for instant disaster recovery using an Elasticsearch plugin if it needs to be verified listings are never offline. Because the Gateway can function as an [S3 Repository](#), you can leverage the [AWS Cloud Plugin](#) to get [Snapshot and Restore](#) capability for your search data (index). To *snapshot* is to back up your search data to a file system or S3 (Swarm); to *restore* is to place a snapshot back into production.

These are key reasons for using the AWS Cloud plugin:

- **Search Index Restoration:** If your Search cluster has problems and the search index is lost, you can restore a snapshot so applications that depend on listings and collections are not interrupted.
- **Usage Snapshot:** The usage metering indices written are temporary. To preserve data being written since the last backup, you can set up frequent snapshots.
- **Data Move:** If you are making changes to your Search cluster, you can restore a snapshot to the new location to minimize disruption in services.

Important

[Refresh the feed](#) for the restored index, and allow time for Swarm to verify the index data. Until it completes, any objects created, changed, or deleted after the last snapshot may be missing or appear erroneously.

Configuring the Plugin

These are the required:

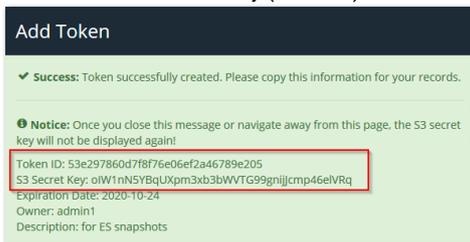
Elasticsearch to backup	<code>http://<elasticsearch-node>:9200</code>
Content UI (Portal)	<code>http://<domain>/_admin/portal/</code>
Domain	<domain> in destination Swarm storage cluster

Bucket	<bucket> within the <domain>
S3 Endpoint	http(s)://<domain>:<S3-port>
Token ID (access key)	UUID
S3 Secret Key	generated when token is created

Best practice

Although you can back up Elasticsearch to the same Swarm cluster that is using it, it is best to use a separate Swarm cluster.

1. In the Content UI (Portal) on the Swarm cluster storing the Elasticsearch snapshots, create an S3 token.
 - a. Create or select the domain.
 - b. Open its **Settings** (gear icon) and select the **Tokens** tab.
 - c. Create a token that includes an S3 key.
 - d. Record *both* the access key (token ID) and the secret (S3) key:



2. On *each* node in your Elasticsearch cluster, install the [AWS Cloud Plugin](#):

- a. Log in to the node as the root user using ssh.
- b. Install the plugin:

```
sudo /usr/share/elasticsearch/bin/elasticsearch-plugin
```

- c. To allow the keys to be specified, set the following JVM option (/etc/elasticsearc/jvm.options):

```
-Des.allow_insecure_settings=true
```

- d. Restart the Elasticsearch service.
3. Configure an S3 repository using the token (see below).
 4. Test the plugin, as shown below:
 - a. Take a snapshot.
 - b. Delete the search index (which causes listings to fail).
 - c. Restore your snapshot using the manifest file.
 - d. Verify the listings are working again.

Configure the S3 Repository

In these examples, the Elasticsearch repository is using S3 to store the search data snapshots.

1. Create the S3 repository using a command like the following. The **base_path** can be empty, or set if this bucket is the backup destination for multiple Elasticsearch clusters.

- The endpoint *with the bucket in the host* must be accessible from every Elasticsearch node (to verify, run `curl -i http://essnapshots.mydomain.example.com/`). This requires either explicit `/etc/host` entries or wildcard DNS for the domain. If any node fails to contact the endpoint, you must delete the repo with `curl -XDELETE 'http://elasticsearch:9200/_snapshot/myRepo'` and PUT it again.
- These configuration values (`endpoint`, `access_key`, `secret_key`) can be stored in `elasticsearch.yml` instead of the JSON body (see the Elasticsearch docs for the config names).

```
curl -XPUT -H 'Content-type: application/json'
  'http://elasticsearch:9200/_snapshot/myRepo'
  -d '{
    "type": "s3",
    "settings": {
      "bucket": "essnapshots",
      "region": null,
      "endpoint": "http://mydomain.example.com/",
      "protocol": "http",
      "base_path": "myswarmcluster",
      "access_key": "18f2423d738416f0e31b44fcf341ac1e",
      "secret_key": "BBgPFuLcO3T4d6gumaAxGalfuICcZkE3mKliwKks"
    }
  }'
```

2. List information about the snapshot repository:

```
curl -XGET
  'http://elasticsearch:9200/_snapshot'
```

3. Verify the repository is created successfully:

```
curl -XPOST
  'http://elasticsearch:9200/_snapshot/myRepo/_verify'
```

Creating a Snapshot the S3 Repository

1. Create a new snapshot into S3 repository, setting it to wait for completion:

```
curl -XPUT
  'http://elasticsearch:9200/_snapshot/myRepo/snapshot_20201031
  ?wait_for_completion=true'
```

If needed, you can restrict the indices (such as to Search only, if Metrics backups are not needed). See [Elasticsearch documentation](#) for details on restricting indices.

2. Allow several hours for this to complete, especially for an initial snapshot of an Elasticsearch with a lot of large indices.

Restoring from a Snapshot

1. **Best practice** – Always test restoring a backup before needed. Delete the search index in a test or staging environment to simulate a situation where restoring Elasticsearch data is needed:

```
curl -XDELETE 'http://elasticsearch:9200/_all' # do not do this in Production!
```

2. In the Storage UI:

- a. Open **Cluster > Feeds**, open the Swarm search feed, and select **Actions (gear icon) > Pause** to prevent a new index from being created.
- b. Open **Settings > Cluster, Metrics** and temporarily disable Swarm metrics (`metrics.targets` set to nothing) to prevent those indices from being created during restore.

3. Restore the search index, renaming indices if they exist and are locked:

```
curl -XPOST "elasticsearch:9200/_snapshot/myRepo/snapshot_20201031/_restore"  
-H Content-type: application/json  
-d '{  
  "rename_pattern": "(.+)",  
  "rename_replacement": "restored_$1"  
}'
```

4. In the Storage UI:

- a. Open **Settings > Cluster, Metrics** and re-enable Swarm metrics (`metrics.targets` set to its prior value).
- b. Open **Cluster > Feeds**, open your Swarm search feed, and select **Actions (gear icon) > Unpause** to reactivate the feed.

5. Verify the listings are working as before:

```
curl -iL 'http://swarm:80/  
?domains&format=json'
```

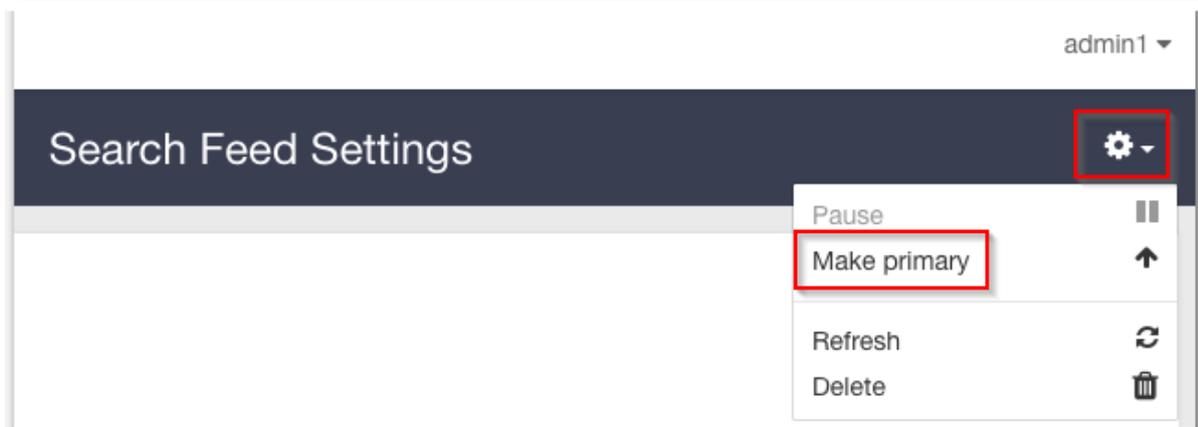
Rebuilding a Search Feed

When the underlying schema for Swarm Search changes, new feeds are required to generate index data in the new format. Swarm Storage allows creation of more than one Search feed so transitioning from using one feed to another is possible without disruption. Continue using the primary feed for queries during the transition; the second feed is incomplete until it fully clears its backlog. Transition to it (marking it as primary) as soon as reasonable for your operations when the second feed is caught up.

Important

Delete the original feed when verifying the new primary feed target is working. Having two feeds is for temporary use only because every feed incurs cluster activity, even when paused.

1. [Create a new search feed](#) in the Swarm UI. *Do not select **Make primary**.*
2. Wait until the new feed has completed indexing the cluster, when the feed shows 0 "pending evaluation".
3. Make it the primary feed when the new feed is ready. Navigate to **Cluster > Feeds**, open the new Search feed, and select **Make primary** from the drop-down menu in the Swarm UI.



4. Operate with both feeds for several days. You can restore the old feed to be primary during troubleshooting if there is a problem.
5. Delete the old feed after this confirmation period. Navigate to **Cluster > Feeds**, open the old Search feed, and select **Delete** from the drop-down menu in the Swarm UI.
6. Delete the old index data to reclaim that space if desired.

```
curl -XDELETE 'http://old-elasticsearch:9200/_all' # do not do this to your production data
```

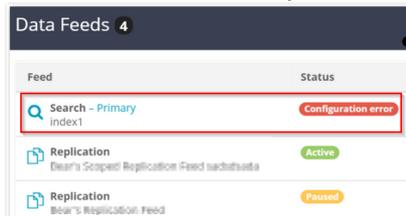
Monitoring Elasticsearch

- [Swarm UI Monitoring](#)
- [Checking ES Cluster Health](#)
- [Diagnosing and Fixing Split Brain](#)

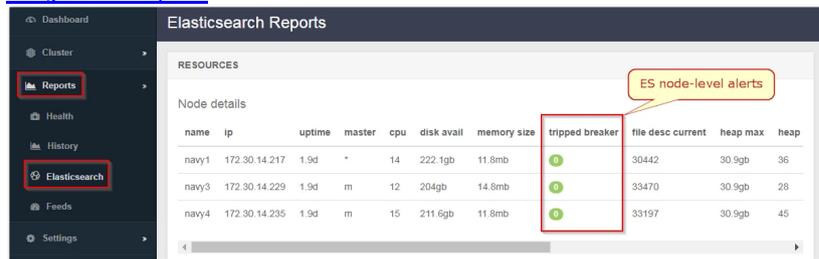
Swarm UI Monitoring

Make it part of a routine to check the [Swarm UI](#) to monitor for problems with an Elasticsearch cluster.

- Navigate to **Cluster > Feeds** to monitor the real-time status of the search feed. Examine the health of the Elasticsearch cluster if the Swarm search index status is yellow or red. See [Search Feeds](#).



- Navigate to **Reports > Elasticsearch** to view details and alerts. See [Elasticsearch Reports in Using Cluster Reports](#).



Che

A first step to any investigation is to query one of the Elasticsearch nodes for a report on the health of the Elasticsearch cluster:

1. Wait until *all* nodes are back in service before proceeding if any Elasticsearch nodes are temporarily out of service for a known reason (such as a reboot or a rolling upgrade), .
2. Query the health of the cluster against one of the Elasticsearch nodes:

```
curl -X get <ES_Server>:9200/_cat/health?v
```

3. Verify the value of **node.total** matches the expected number of nodes in the Elasticsearch cluster.

Diagnosing and Fixing Split Brain

A split brain situation is created when one or more nodes fails in a cluster and the cluster reforms itself with the available nodes. Believing the other clusters are dead, each cluster may simultaneously access the same data, which can lead to corruption.

1. Perform a health check of the ES cluster (see above).
2. Perform the same health query against each of the other Elasticsearch nodes in the cluster if **node.total** is less than expected.
3. Elasticsearch cluster is experiencing a split-brain situation if different Elasticsearch nodes report different values for **node.total**.
4. Examine the `/etc/elasticsearch/elasticsearch.yml` configuration files and verify the Elasticsearch nodes are all configured correctly. Contact DataCore Support if help is needed to verify these settings.
5. Examine the value of the "unassign" shards. There may be a shard allocation issue that is causing the Elasticsearch cluster to have a non-green status if the value is greater than zero. Contact DataCore Support for help in resolving this situation.

Adding Nodes to an ES Cluster

VM users when cloning ES servers

Before starting ES on the new (cloned) node, delete all data under the configured data location for the cloned node (e.g., `/var/lib/elasticsearch`). ES generates an error stating a conflicting node store cannot be used if the data is not cleared out before ES on the cloned node is started.

The symptom of this condition is the ES service shows as running per `systemd` and the network table (`netstat`) shows ES listening on ports 9200 and 9300, but any connection to port 9200 on the cloned ES node is refused.

Complete these steps to add a new node to a running Elasticsearch cluster:

1. Install the new ES server.
 - a. Verify the new server meets the prerequisites in [Preparing the Search Cluster](#).
 - b. From the Swarm bundle download, get the latest Elasticsearch RPM and Swarm Search RPM, which installs plugins and support utilities.

```
elasticsearch-VERSION.rpm
caringo-elasticsearch-search-VERSION.noarch.rpm
```

- c. Install the Caringo RPM public key included with the distribution bundle by running the following command:

```
rpm --import RPM-GPG-KEY
```

- d. Install the RPMs.

```
yum install elasticsearch-VERSION.rpm
yum install caringo-elasticsearch-search-VERSION.noarch.rpm
```

2. [Configure the ES server](#) using the installation script: `/usr/share/caringo-elasticsearch-search/bin/configure_elasticsearch_with_swarm_search.py --no-bootstrap`
 - a. Install as if this was the first of x ES servers, where x is how many ES servers exist in the ES cluster.
 - b. Choose **No** when it prompts to start ES services.
 - c. It prompts for information on all other ES nodes, and it creates configuration files for each. Save these configuration files, which are useful for any future redeployment.
3. In the Swarm UI, pause the [Search feed](#).
4. Stop the ES services on each of the existing nodes.
5. SSH into each existing node and edit `/etc/elasticsearch/elasticsearch.yml`:

- a. Add a comma and the new ES server to the list for `discovery.seed_hosts`.

```
discovery.seed_hosts: ["es1.example.com", "es2.example.com", "es3.example.com", "NEW-ES-N
```

The equivalent ES 6.8.6 config was named `discovery.zen.ping.unicast.hosts` and required setting `discovery.zen.minimum_master_nodes` to be $(\text{total number of nodes})/2 + 1$ but is no longer necessary with ES 7.

- b. Set to the new number of nodes in the ES cluster for `gateway.expected_nodes`.
- c. Adjust the value as appropriate for `gateway.recover_after_nodes`. This is the minimum number of running ES nodes before going in to the operation status:
 - Set to 1 if total nodes are 1 or 2.
 - Set to 2 if total nodes are 3 or 4.
 - Set to the number $- 2$ if the total nodes are 5 to 7.
 - Set to the number $- 3$ if total nodes 8 or more.

6. Add the new server to the syslog on the central logging host.
7. Start the ES services on all ES servers.
8. Check the status to verify they all show the correct number of nodes and have a status of green.

```
curl [ES-NODE-IP]:9200/_cluster/health?pretty
```

9. In the Swarm UI, resume the [Search feed](#).

Resetting Elasticsearch

Perform a reset if the state of Elasticsearch and Historical Metrics needs to be cleared (such as if an index is deleted and erroneously recreated without the Swarm schema). A reset is a method to delete an index and refresh a feed safely, by deleting feeds before removing index data.

1. Set Swarm's configuration setting `metrics.target` to blank (which can be performed using SNMP or REST).

```
curl -i -u admin:PASSWORD -XPUT "http://SWARM·NODE:91/api/storage/clusters/CLUSTER·NAME/settings/metrics.target"
```

2. Delete the current Search Feed definition in the Swarm UI.
3. Delete the Swarm search index.

- a. Use the following command to determine the Swarm search index:

```
curl http://ES·NODE:9200/_cat/indices | grep 'index_SWARM·CLUSTER·NAME'
```

- b. Run the following command to delete the Swarm search index:

```
curl -X DELETE http://ES·NODE:9200/SWARM·SEARCH·INDEX
```

4. Delete historical metrics indices as needed.

The following command deletes *all* metrics indices. Verify the glob pattern used to avoid deleting needed indices.

```
curl -X DELETE 'http://ES·NODE:9200/metrics-SWARM·CLUSTER·NAME-*
```

5. Create a new Search Feed definition pointing to the ES servers in the UI (Swarm UI or legacy Admin Console). (This step creates the feed using the Swarm schema.)
6. Reinitialize the curator:

```
/usr/share/caringo-elasticsearch-metrics/bin/metrics_curator -n -v
```

7. Reset the Swarm setting `metrics.target` back to the correct value (which can be performed using SNMP or REST).

```
curl -i -u admin:PASSWORD -XPUT --data-binary '{"metrics.target":"METRICS·HOST"}' \
http://SWARM·NODE:91/api/storage/clusters/CLUSTER·NAME/settings/metrics.target"
```

Rolling Restart of Elasticsearch

Elasticsearch immediately begins regenerating metadata (reallocating those shards) when an ES nodes becomes unresponsive. In controlled situations such as a rolling restart steps can be taken to minimize the impact.

A rolling restart of the ES cluster may be needed to upgrade the Elasticsearch version or to perform maintenance on the server itself such as an OS update or hardware change. This process keep the cluster operational while taking nodes offline one at a time. Because Elasticsearch prefers to keep data fully replicated and evenly balanced, it must be made to pause rebalancing until the rolling restart completes.

This pausing is performed through ES settings changes, as recommended by Elasticsearch in the bolded links below. The essential process is this:

1. Start maintenance mode by changing the ES settings.
2. Complete the maintenance work (such as upgrading ES) and rolling restart.
3. Stop maintenance mode by restoring the ES settings.

Follow the steps in the Elasticsearch documentation, [with this important addition for versions before Elasticsearch 7:](#)

1. Set the `discovery.zen.minimum_master_nodes` value in the `/etc/elasticsearch/elasticsearch.yml` config file to be: $(\text{number of master-eligible-nodes}/2, \text{rounded down}) + 1$ when [the elasticsearch cluster is first configured](#). Verify the number is *smaller than the number of currently available nodes*. Run the following command now to set that number to be `<current_min_master_nodes>`: $(\text{current number of master-eligible-nodes}/2, \text{rounded down}) + 1$ if not.

```
curl -s -XPUT 'http://ES_NODE:9200/_cluster/settings' \
  --data-binary '{"transient": {"discovery.zen.minimum_master_nodes" : "<current_min_master_n
```

2. Follow the Elasticsearch rolling restart procedure for the version:

- [ES Rolling Upgrades \(current\)](#)
- [ES 5.6 Rolling Upgrades](#)
- [ES 2.3.3 Rolling Restarts](#)

3. Reset `discovery.zen.minimum_master_nodes` to its original value, or adjust it based on the current number of expected available nodes after completing the rolling restart:

```
curl -s -XPUT 'http://ES_NODE:9200/_cluster/settings' \
  --data-binary '{"transient": {"discovery.zen.minimum_master_nodes" : "<original_min_master_n
```

Uninstalling Elasticsearch

If you need to uninstall Elasticsearch for any reason, pause or delete the search feed before stopping the Elasticsearch service.

To uninstall the Search service:

1. Pause or delete the search feed. (See [Managing Feeds.](#))
2. Log in as root on the Elasticsearch server.
3. Stop the Elasticsearch service and uninstall the service:

```
yum remove caringo-elasticsearch-search
yum remove elasticsearch
```

This does not erase the packages, so the packages can be restored later, if needed.

4. The Metrics curator relies on the Elasticsearch service, so do one of the following:
 - Reconfigure Swarm Metrics (`/etc/caringo-elasticsearch-metrics/metrics.cfg`) to point to a different ES cluster (or remove `cluster = <cluster-name>`)
 - Uninstall Swarm Metrics

```
yum remove caringo-elasticsearch-metrics
yum remove elasticsearch-curator
```

Merging and Renaming ES Clusters

This technique allows merging two separate Elasticsearch clusters or renaming an Elasticsearch cluster. This is performed by retiring an Elasticsearch node "into" a new Elasticsearch cluster.

1. Join one or more existing nodes to the new cluster by changing the `cluster.name` value to the new cluster's name.
2. Verify the join.
3. Decommission a node by telling the cluster to exclude it from allocation to migrate shards off of the old nodes.

```
curl -XPUT localhost:9200/_cluster/settings -d '{
  "transient" : {
    "cluster.routing.allocation.exclude._ip" : "10.0.0.1"
  }
};echo
```

This causes Elasticsearch to allocate the shards on that node to the remaining nodes, which is performed without the state of the cluster changing to yellow or red (even replication 0).

4. Shut down the node when all shards are reallocated.
5. Include the node for allocation to restore the node to service, which causes Elasticsearch to rebalance the shards again.

See the [Elasticsearch documentation](#).

Swarm Storage Cluster

This section explains essential Swarm concepts and describes how to configure and manage a Swarm Storage cluster.

For details on using the Swarm UI for Swarm administration through a browser, see [Swarm Storage UI](#).

Proactive support

Swarm clusters send basic configuration, usage, and health information reporting to DataCore once a day from every node in the cluster. No user-stored data is included, and this information is encrypted prior to transmission. Contact your support representative with any questions related to this support functionality.

- [Managing Domains](#)
- [Configuring Swarm Storage](#)
- [Prometheus Node Exporter and Grafana](#)
- [Swarm Storage Policies](#)
- [Managing and Optimizing Feeds](#)
- [Swarm Concepts](#)
- [Defining Swarm Admins and Users](#)
- [Managing Volumes](#)
- [Using SNMP with Swarm](#)
- [Troubleshooting Storage](#)

Managing Domains

This section provides information about domains and describes how to manage them in a Swarm storage cluster.

A *domain* is a secure realm defined for controlled access and administration; it lives entirely within the Swarm storage cluster. *Unnamed objects* reside at the root of the domain.

Like a storage facility containing many storage units, a domain may contain multiple *buckets*, for holding *named objects*. Buckets allow named objects to be grouped by usage or type (such as documents, photos, and videos) and to control access to each group.

See [Configuring Domains](#) for domain management from a browser using the Swarm Content UI.

- [Renaming Domains and Buckets](#)
- [Restoring Domains and Buckets](#)
- [Guidelines for Managing Domains](#)
- [Recreating Buckets](#)
- [Resolving Duplicate Domain Names](#)
- [Accessing Inaccessible Objects with CID](#)
- [Manually Creating and Renaming Domains](#)

Renaming Domains and Buckets

If duplicate domains or buckets are involved, rename them using the correct syntax. Rename context objects (domains and buckets) in a storage cluster using the [SCSP COPY](#) command with the **newname** [query argument](#) that specifies the new name.

Naming and Slashes – Follow the [Naming Rules](#) for domains and buckets. Swarm handles slashes:

- *Leading slashes* (/εοο) are silently removed in all cases.
- *Trailing slashes* (εοο/) are silently removed for buckets, but they cause *404 Page Not Found* errors for domains. (v11.1)



Tip

Log in to the Content UI to verify the renaming.

Preserving headers

Copy existing [headers](#) to be preserved.

- In particular, look for **Policy-*** headers and the **x-tenant-meta-name** header that Gateway uses to group domains under a tenant.
- Add the **preserve** [query argument](#) to the COPY request to verify any custom metadata existing on the object is carried over to the copy. (v9.2)
- To overwrite an existing value, include the header name with the new value on the request.

See *Headers to preserve* in [SCSP COPY](#).

Renaming a Domain

If a duplicate domain is found, rename the domain.

```
curl -i --location-trusted -u admin -X COPY
  'http://{host}?domain={old-name}&admin&newname={new-name}&preserve'
[-D log-file-name]
```

Example

```
curl -i --location-trusted -u admin -X COPY
  'http://172.16.0.35?domain=abc.example.com&admin&newname=xyz.example.com&preserve'
[-D log-file-name]
```

The message **New object created** confirms that the renaming procedure was successful.

Renaming a Bucket

If a duplicate bucket is found, rename the bucket.

```
curl -i --location-trusted -u admin -XCOPY
  'http://{host}/{old-name}?domain={domain}&admin&newname={new-name}&preserve'
[-D log-file-name]
```

Example

```
curl -i --location-trusted -u admin -XCOPY
  'http://172.16.0.35/oldbucketname?domain=abc.example.com&admin&newname=newbucketname&preserve'
[-D log-file-name]
```

The message **New object created** confirms that the renaming procedure was successful.

Restoring Domains and Buckets

- [Recovering a Deleted Domain](#)
- [Recovering a Deleted Bucket](#)

Recover a bucket by recreating the object using the **recreatecid** query argument if a domain or bucket (context object) is deleted in the storage cluster by mistake.

 **Caution**

Any mistakes in using these commands can cause serious problems. Consult DataCore Support for assistance with these operations. Do not attempt to use **recreatecid** to move bucket contents across domains within the cluster.

 **Tip**

If the grace period ([health.recursiveDeleteDelay](#)) following the recursive deletion of a domain or bucket, use the special methods below to restore it without data loss. If the deletion had no grace period (**recursive=now**), only some of the data is lost to reclamation, depending on how much time has passed since the delete.

Recovering a Deleted Domain

A new domain cannot be created with the identical name to recover a domain, because it is mapped to a new UUID; **Castor-System-Alias** on the domain stream. Create a new domain from the command line and use a query argument to apply the previous domain's UUID.

Orphaned buckets - The buckets in the deleted domain reference are not by name but by UUID (**Castor-System-CID**). These buckets are not accessible until a new domain is created which uses the original UUID.

To recover the domain:

1. Locate and record the log message related to the missing domain.

```
Domain 'example.com' (uuid=a2fc4bb0fc31bbc73a088783aef8ea73) has been deleted ...
```

2. Copy the UUID listed within the log message to recreate the missing domain.

```
a2fc4bb0fc31bbc73a088783aef8ea73
```

3. Create a new domain with a POST that references the deleted domain's UUID in the **recreatecid** query argument:

```
curl -i -X POST --location-trusted --post301 --user 'admin:datacore' --data-binary '' \
  -H 'Content-type: application/castorcontext' \
  'http://10.160.132.33?domain=unwanted-chs-eu-domain1&admin&recreatecid=04648a25e90c0b0364:
```

4. If the domain and its content are not required, delete them using:

```
curl -i --location-trusted -u admin:datacore -X DELETE
'http://10.160.132.33/04648a25e90c0b036435caa6d03295be?admin&recursive'
```

Recovering a Deleted Bucket

The named objects within the bucket are inaccessible until the bucket is recovered after deleting a bucket.

Warning

Do not use `recreatecid` as a method to move bucket contents across domains within the cluster; this causes critical errors.

To recover the bucket:

1. Locate and record a critical log message related to the missing bucket, resulting from a named object within it being inaccessible.

```
Bucket 'mybucket' (uuid=75edd708dc250137849bbf590458d401) in domain 'example.com' has been c
Consider recreating.
```

2. Copy the UUID listed within the log message to recreate the missing bucket.

```
75edd708dc250137849bbf590458d401
```

3. Create a new bucket with a POST that references the deleted bucket's UUID in the **recreatecid** query argument:

```
$ curl -i -X POST --location-trusted --post301 --anyauth --user 'admin:password' --data-bin:
-H 'Content-type: application/castorcontext' \
-H "Policy-*: {if needed}" \
'http://{host}/mybucket?domain=example.com&admin&recreatecid=75edd708dc250137849bbf590458d'
```

Guidelines for Managing Domains

Follow these guidelines when creating domains:

- **Set up a default cluster domain** (a domain name that exactly matches the name of the cluster). Every object with no domain explicitly defined belongs to the default cluster domain.
- **Create at least one domain for named objects.**
- **Verify all domain names are unique** among all managed clusters. An SCSP operation to create a domain with the same name in multiple storage clusters creates different domains that share the same name. This leads to name collision and incorrect results if the different domains are replicated into the same cluster. Create the domain name once and use Swarm remote replication to copy it into separate clusters when creating a new domain.
- **Verify all domain names are [IANA-compliant](#)** (`cluster.example.com`). Create an IANA-compliant domain name and then create all named objects in buckets in that domain if a cluster name that is not IANA-compliant currently exists. See [Naming Rules](#).

Recreating Buckets

It is possible to delete a bucket and recreate another bucket with the same name. The new bucket is a different bucket (a different ID) with the same name.

All objects in a bucket are inaccessible after deletion, even if another bucket with the same name is subsequently created.

i Best practice
For best results, wait at least *twice* the value of the `cache.realmStaleTimeout` parameter before recreating a bucket with the same name as a bucket deleted.

Wait 20 minutes after deleting a bucket with the [cache.realmStaleTimeout](#) value of 10 minutes (600 seconds) before creating the new bucket.

Resolving Duplicate Domain Names

- [Manually Renaming a Domain or Bucket in a Mirrored or DR Cluster](#)
- [Example of renaming a domain](#)

Verify all domain names and all bucket names within a particular domain are unique among all managed clusters when creating domains. This is important when one cluster replicates to another. Create two types of DR cluster configurations using Swarm Feeds:

- **DR Cluster** - Copies one or more clusters and content in another physical location.
- **Mirrored Configuration** - Copies content of cluster 1 to 2 and the contents of cluster 2 to 1.

In either type of configuration, if two clusters contain two unique domains/buckets with the same external name, feeds replication creates a duplicate domain/bucket name(s) in the DR or mirrored cluster. This results in indeterminate access to objects in the duplicated domains/buckets. Sometimes a request to a particular object in one of the duplicate domains/buckets succeeds, but other times it fails.

Swarm logs a Critical error to the Admin Console similar to the following when it detects a duplicate:

```
SCSP CRITICAL: Domain 'collisiondomain.e0f55af9abcacd625cfd946a1a5e49d0'
(uuid=15748c61aea50ec3bcdd28df763f6cfa) has collided with existing Domain
(uuid=6ba3aeda10f2254e5b418b73c684c838).
Remove or rename one of the versions to avoid conflict.
```

Perform one of the following procedures if a Critical error is received:

- **Manually rename a duplicate domain in the replication source cluster.** Admin Console is legacy and unavailable in most installs wherever `security.noauth=true`, the default for a while.

Important

This method does not work in a mirrored configuration because both clusters have duplicates. In this situation, use the next procedure (Manually rename either conflicting domain or bucket in either cluster)

- **Manually rename either conflicting domain or bucket in either cluster.** This method is used in a mirrored cluster conflict. It resolves the issue and prevents it from reoccurring. This method is not recommended for a DR cluster conflict because the next time the same domain or bucket is replicated to the DR cluster from the source, the duplicate domain name still exists.

Manually Renaming a Domain or Bucket in a Mirrored or DR Cluster

Use the [SCSP COPY](#) command with the following query arguments and authenticate as a cluster administrator to manually rename a domain or bucket in a mirrored or DR cluster.

Query Argument	Meaning
<code>admin</code>	Also called as <i>administrative override</i> , this query argument allows ignoring <code>Allow</code> headers and bypass the <code>Castor-Authorization</code> header. Note <ul style="list-style-type: none"> • Must use this query argument with cluster admin credentials. • Administrative override does <i>not</i> affect lifepoint policy deletability for immutable objects.
<code>newname=new-domain-name</code>	The new name for the domain or bucket. See Guidelines for Managing Domains .
<code>oldname=domain-UUID</code>	The UUID of the domain or bucket to rename. The UUID can be found in the critical log message printed when a duplicate is detected.

To rename a domain in a mirrored or DR cluster:

1. Record the old name for the duplicate domain or bucket to be renamed from the related critical error message.
2. HEAD the old name of the domain or bucket to see if it has protection settings that need to be modified.
Use the below command:

```
curl --head --location-trusted --anyauth -u 'admin:datacore'
'http://{swarm-host}/mybucket?domain=cluster.example.com&admin&oldname=bbc2365b3283c23c4759!
-D rename-domain.log
```

3. Authentication as a cluster administrator is required (a user in the `security.administrators` parameter).
4. Rename the domain or bucket.

```
curl -i -X COPY
--anyauth -u 'admin:ourpwdofchoicere'
--location-trusted
'http://{swarm-host}/mybucket?domain=cluster.example.com
&admin
&oldname=bbc2365b3283c23c47595abcf09034a
&newname=mynewbucket
&preserve'
-D rename-domain.log
```

5. Drop and recreate the search feed to update the new name in the search and listing requests (see [Search Feeds](#)).

Example of renaming a domain

Rename the **cluster.example.com** domain to **archive.example.com** by sending commands to a node whose IP address is 172.16.0.35:

1. Record the old name for [cluster.example.com](#) from the related critical error message (e.g., `bbc2365b3283c23c47595abcf09034a`).
2. HEAD the domain to get the protection settings if any.

```
curl -i
  --anyauth -u 'admin:ourpwdofchoicehere'
  --location-trusted 'http://{swarm-host}/bbc2365b3283c23c47595abcf09034a?admin'
```

Sample output:

```
HTTP/1.1 200 OK
Cache-Control: no-cache-context
Castor-Authorization: cluster.example.com/_administrators, POST=cluster.example.com
Castor-Stream-Type: admin
Castor-System-Alias: bbc2365b3283c23c47595abcf09034a
Castor-System-CID: ffffffffffffffffffffffffffffffffff
Castor-System-Cluster: cluster.example.com
Castor-System-Created: Wed, 17 Nov 2010 15:59:13 GMT
Castor-System-Name: cluster.example.com
Castor-System-Owner: admin@CASTor administrator
Castor-System-Version: 1290009553.775
Content-Length: 0
Last-Modified: Wed, 17 Nov 2010 15:59:13 GMT
lifepoint: [] reps=16
Etag: "099e2bc25eb8346ed5d94a598fa73bfa"
Date: Wed, 17 Nov 2010 16:02:07 GMT
Server: CASTor Cluster/5.0.0
```

3. Update a **Castor-Authorization** header to reflect the new name for the duplicate if it is present; otherwise, there is no need to add it. The information needed:

```
Castor-Authorization: cluster.example.com/_administrators, POST=cluster.example.com
```

This header must be changed to:

```
Castor-Authorization: archive.example.com/_administrators, POST=archive.example.com
```

The following headers must be added *exactly* as shown:

```
-H 'Castor-Stream-Type: admin'
-H 'lifepoint: [] reps=16'
```

lifepoint: [] reps=16 enables the domain to be replicated as many times as possible.

Use **Castor-Stream-Type: admin** for all objects that use a **Castor-Authorization** header.

4. Rename the domain.

```
curl -i -X COPY
  -H 'Castor-Authorization: archive.example.com/_administrators, POST=archive.example.com'
  -H 'Castor-Stream-Type: admin'
  -H 'lifepoint: [] reps=16'
  --anyauth -u 'admin:ourpwdofchoicehere'
  --location-trusted
  'http://{swarm-host}?domain=cluster.example.com
  &admin
  &oldname=bbc2365b3283c23c47595abcf09034a
  &newname=archive.example.com'
  -D rename-domain.log
```

5. Drop and recreate the search feed to update the new name in the search and listing requests (see [Search Feeds](#)).

Accessing Inaccessible Objects with CID

A domain, bucket, or named object is accessible by ID using a *Context Identifier (CID)* query argument if it is inaccessible by name. The CID query argument syntax is: **cid=**

This troubleshooting process is helpful when:

- The domain or bucket was deleted.
- The domain was duplicated in a disaster recovery cluster (two domains exist with the same name in the same cluster). (See [Resolving Duplicate Domain Names.](#))



Note

This procedure allows access to the object but not to recover it. To recover accidentally deleted domains and buckets, see [Restoring Domains and Buckets.](#)

The value of the object's **Castor-System-CID** header identifying the object's parent is needed to execute a **cid=** query argument. Locate the value of the Castor-System-CID header if an object named `photo1.jpg` is not accessible.

This information can be located in one of two ways if it was not recorded or stored:

- **Review the debug-level system logs.** These logs record the Castor-System-CID value every time you access the object.
- **Use the Content Router Enumerator.** (Swarm 7.2 or prior only) This tool iterates through all objects in a cluster and returns information about each object.

To implement this tool:

1. Add a Content Router filter rule to search for objects where the value of the **CastorSystem-Name** header is the name of the inaccessible object.
2. Instantiate a metadata enumerator subscribed to the rule channel you created in the preceding bullet to obtain the object's metadata using the SDK.
3. Look for the value of the **Castor-System-CID** header in the metadata returned for the object.

See the *Enumerator setup* in the [SDK Overview.](#)

Access the object using the **cid=CID-header-value** query argument after locating the value of the object's **Castor-System-CID** header. Enter the following URL in the browser's address or location field to access a named object using a web browser:

```
http://node-ip/object-name?cid=CID-header-value
```

Enter the following to access an object named `file.html` with a CID of `55aba17ad53c61782d7dd0afa8dd2f7d`:

```
http://node-ip/file.html?cid=55aba17ad53c61782d7dd0afa8dd2f7d
```

Manually Creating and Renaming Domains

- [Best practices](#)
- [SDK for Creating a Domain](#)
- [cURL for Creating a Domain or Bucket](#)
 - [Create a domain](#)
 - [Create a bucket](#)
- [Best practice](#)

New domains can be created in a storage cluster in three ways:

1. [Swarm Content UI](#) (preferred); see [Configuring Domains](#)
2. Programmatically, using Swarm SDK
3. Manually, using [cURL](#)



Best practices

- Use the [Content UI](#) to create each new context (domain or bucket), because the UI automatically creates the corresponding domain managers and adds the correct protection settings to a cluster. Contact the cluster administrator to access the Content UI and create a new context if a new domain is needed.
- Do not attempt to create domains and buckets manually unless the administrator is unavailable, Content UI access is unavailable, *and* have advanced user with experience in creating domains and buckets.
- Create [Content Policies](#) (protection settings) for domains and buckets matching the settings normally created by the Content UI. Contact DataCore Support for help manually enabling the protection settings.
- Check the [Naming Rules](#) before creating any context objects. Unlike cluster names, domain and bucket names *cannot include spaces*.

SDK for Creating a Domain

The Swarm SDK version 1.4 and later includes classes assisting in creating a domain. The following table lists the classes and corresponding source code location in the SDK distribution.

See the [SDK Overview](#).

Class	Location
C++	<code>sdk-extract-dir/cpp/src/realm</code>
C#	<code>sdk-extract-dir\csharp\ScspCSExamples\ScspRealmExamples.cs</code>
Java	<code>CASTorSDK-src-extract-dir/com/caringo/realm</code>
Python	<code>castorsdk-python-egg-extract-dir/castorsdk/realm</code>

cURL for Creating a Domain or Bucket

The example below shows how to create a domain with no domain protection setting, allowing any user to POST to the domain. For guidance on authentication, see [Content Gateway Authentication](#).

To create the domain manually, use the following syntax. Either the **domain** query argument or a **Host** header is required, even if the domain being created is the default cluster domain (see [Types of Container Objects](#)) because SCSP methods on a domain are executing.

Create a domain

```
$ curl -i -X POST --location-trusted --post301 --anyauth --user 'admin:password' --data-binary ''
-H 'Content-type: application/castorcontext' \
-H "Policy-*: {if needed}" \
'http://{host}/?domain=newdomain.example.com'
```

Create a bucket

```
$ curl -i -X POST --location-trusted --post301 --anyauth --user 'admin:password' --data-binary ''
-H 'Content-type: application/castorcontext' \
-H "Policy-*: {if needed}" \
'http://{host}/newbucket?domain=mydomain.example.com'
```

Headers

- **Content-type: application/castorcontext** – specifies creating a domain or bucket, and is required when sending requests to the Gateway (see [Domain and Bucket Creation](#)). It replaces the **createdomain** query argument, which is deprecated. The Swarm setting [scsp.requireExplicitContextCreate](#) protects objects from being created erroneously as contexts (buckets or domains); with this setting enabled, Swarm does not create a context object unless it includes this header. (v9.1)
- **Policy-*** headers – add for any domain-specific [replication](#), [ec-encoding](#), and [versioning](#) requirements. See [Configuring Cluster Policies](#).

Swarm returns a **201 Created** response with the result on success: `New stream created.`



Best practice

Add conditional headers to verify updates are being made to the intended domain or bucket.

Use `if-none-match` to prevent creating a domain or bucket that already exists:

```
$ curl -X PUT http://{host}/_admin/manage/tenants/_system/domains/mydomain
-H "if-none-match: *"
```

Use `if-match` on the ETag to prevent updating the wrong domain or bucket:

```
$ curl -X PUT http://{host}/_admin/manage/tenants/_system/domains/mydomain
-H "if-match: \"ETAG\""
```

Configuring Swarm Storage

This section describes how to configure a Swarm storage cluster.

To configure a new cluster, modify the configuration (`.cfg`) file according to the implementation type:

- A cluster uses a `cluster.cfg` if the cluster boots from CSN or Platform Server.
- Prepare a `node.cfg` and boot one of the following ways if the cluster does not boot from CSN or Platform Server:
 - PXE boot the cluster
 - Boot the cluster using a centralized configuration server
 - Boot from a USB flash drive (requires an update to the `caringo/node.cfg` file in each node with identical settings)



Note

Because of the variety of ways clusters can be configured, the configuration file is referred to generally as the *cluster/node config*. Understand it to mean the CFG file where Swarm accesses settings for a *specific individual* cluster.

- [Managing Configuration Settings](#)
- [Settings Reference](#)
- [Configuring an Rsyslog Server](#)
- [Configuring the Overlay Index](#)
- [Configuring Encryption at Rest](#)
- [Configuring Cluster Policies](#)
- [Configuring Volumes Options](#)
- [Configuring an External Time Server](#)
- [Configuring External Logging](#)
- [Time of Last Access - atime](#)
- [Configuring Power Management](#)
- [Configuring Content Integrity Settings](#)

Managing Configuration Settings

- [Scope of settings](#)
- [Qualified versus unqualified settings](#)
- [Organizing settings by section](#)
- [Formatting settings](#)
- [Upgrading a configuration file](#)

Sections. The option exists to organize cluster settings in to *sections*. A configuration file section is identified by a unique name within square brackets (**[section-name]**) containing settings logically related to each other.

Guidelines: Follow these guidelines to manage configuration files in storage clusters:

1. Set the appropriate configuration settings.
2. If upgrading, add, remove, or adjust the settings and values as directed.
3. Use new and empty sections properly.

Scope of settings

Each setting has one of two scopes:

- **cluster-wide:** Used across the entire cluster. This is the default scope, so it is not specified in the reference. Every cluster-wide setting must be identical for all nodes in the node or cluster configuration file.



Warning

Any difference in the *cluster-wide* configuration settings among the nodes can cause serious errors.

- **node-specific:** Specific to each node (chassis or virtual machine). These are the exceptions to the majority (cluster-wide), so they are flagged in the reference. Each cluster node can have a different value.

Qualified versus unqualified settings

Swarm supports both qualified and unqualified configuration settings.

- An **unqualified** setting (such as `corporate`) must be contained in the designated [section], in the empty [] section, or in a configuration file with no sections (or the empty section).
- A **qualified** setting (such as `networks.corporate`) is the name of the section, a period, and the unqualified setting name.

A configuration file can have a mixture of unqualified and fully qualified settings, provided the settings are in the proper sections or the configuration file has no sections (or the empty section).

Organizing settings by section

In Swarm, sections are optional (unless using new, unqualified names). Sections must be used properly if using them.

New sections

Errors result and Swarm does not start if placing unrelated parameters or an old parameter name in any section. The following parameter is not valid:

```
[health]
misplaced-setting = 900          # Swarm does not start
```

The following setting is valid:

```
[health]
startDelay = 900
```

Empty sections

The empty section [] (empty set of square brackets) is a special container clearing any section that may have preceded it. The following setting is valid:

```
[health]
startDelay = 900
[]
ipAddress = 192.168.0.33
```

Formatting settings

All configuration files contain the name-and-value pairs for setting configuration options.

- The format for these pairs is `name = value`.
- **Whitespace** before the name, between the name and =, and between = and the value is ignored.
- Quotes (") are not stripped from values.
- Blank lines and lines beginning with a hash tag (#) are ignored. If this tag appears within the parameter value (anywhere after =), it is part of the option's value field and not a comment.

 **Important**

Settings and values are **case-sensitive**, and names must have **no spaces**.

 **Warning**

Spelling or case errors or illegal values can prevent the node or cluster from booting.

Upgrading a configuration file

Update a node or cluster configuration file by placing new settings, renamed settings, and sections at the end of the existing configuration file. This prevents conflicts with settings located at the beginning of the file.

Use the following guidelines when adding new settings:

- **Add new configuration settings in sections or a fully qualified format.** Adding new settings at the end of the configuration file avoids problems preventing the cluster from booting. Place it at the end of the node or cluster configuration to add the new encoding setting. The new setting does not affect any settings before it. The cluster boots normally and nothing else in the configuration file needs to be modified.
- **Rename existing settings.** Renamed settings are deprecated and are removed eventually in a future release.
See the [Renamed Settings](#).
- **Add sections to the configuration file, verifying all settings are in the correct sections.** *All* settings after a section placed in the node or cluster configuration file must either be placed in respective sections or must be fully qualified.
See the [Configuration Settings](#).

Settings Reference

- [Cluster Settings](#)
- [Chassis \(Node\) Settings](#)

Following are the published settings allowing configuration of how storage cluster nodes operate. They are sorted alphabetically by section, and some do not appear in the provided `node.cfg.sample` configuration file. Use the [Swarm UI](#) to change these settings.

- See [Renamed Settings](#) to see settings changed from earlier versions.
- See [Alphabetical Settings List](#) to see settings listed alphabetically by setting name (not section).

Dynamic settings and SNMP – Many cluster settings (and a few node settings) are *dynamic*, accepting runtime changes without any hardware restarting. Dynamic settings are those with SNMP names defined below. Settings that cannot be changed dynamically are flagged with a restart icon in the Swarm UI, settings that cannot be changed dynamically are flagged with a restart icon: The changes are applied and take effect after the next restart. See [Persisted Settings \(SNMP\)](#) to change settings by SNMP.

When a setting is not dynamic, it is because changing it on a running cluster can destabilize the cluster; examples are `cluster.name` and `cluster.enforceTenancy` (see [How enforceTenancy Works](#)).

Node (Chassis) settings – Each physical or virtual machine is one Swarm *node*. The local config file is read and the node inherits the settings from that file on reboot. Most of the node (chassis) settings (those without an **SNMP Name**) require a reboot of the affected machine (chassis) to take effect; the Swarm UI flags such settings with a restart icon.

Tip

The special value -1 is often used to direct Swarm to take the value of another setting. Follow the guidance in the setting description to use it correctly in each context.

Cluster Settings

Name (SNMP)	Default	Description & Examples
bidding.relocationThreshold SNMP: relocationThreshold	5	Percentage, 0-100. How much difference between volume utilizations causes a lower bid on another node to relocate or rebalance a replica to the other node. Lower values improve load balancing and throughput. Higher values minimize data movement at the expense of lower maximum throughput.
cip.group SNMP: group	224.0.10.100	The multicast IP address for the cluster, as a Class D IP address in the 224.0.0.0 - 239.255.255.255 range. This address must be unique for each cluster. When configuring multiple, distinct clusters, take care the multicast groups do not overlap, as any node with the same multicast group becomes part of a single cluster. Examples: 224.5.5.7 239.255.255.253
cip.queryRetryMultiplier SNMP: queryRetryMultiplier	1	What multiple of time to wait on each successive UDP multicast read retry.

<p>cip.ttl</p>	<p>1</p>	<p>Controls configuration of multicast network traffic TTL (time to live). When set to 1, the multicast traffic should remain on the subnet.</p>
<p>cluster.enforceTenancy SNMP: enforceTenancy</p>	<p>false</p>	<p>Setting to True (recommended) guarantees all content is written into a domain named in the request or else into the default domain. Setting to False (default) allows backward compatibility for applications in use before Swarm 5.0 that access data outside of domains and is required when using Gateway in legacy only mode to access this kind of content. Set to True for new deployments.</p>
<p>cluster.name SNMP: cluster</p>		<p>The name of the cluster. Use an IANA-compatible domain name, such as cluster.example.com, and create one domain with the same name as the cluster, which sets up a default cluster domain holding all unnamed objects. Do not use spaces in the name. To prevent confusion, configure all nodes in the cluster with the same cluster name. Example: http://swarm1.company.com</p>
<p>cluster.proxyIPAddress</p>		<p>[deprecated] The reverse proxy IP address for the cluster. Use cluster.proxyIPList instead. Example: 129.3.7.14</p>
<p>cluster.proxyIPList SNMP: clusterProxyIPList</p>		<p>For use with bidirectional GET replication only, to configure proxies on the source side for the target nodes to connect to. A comma-separated list of reverse proxy IP addresses or names, including ports in name: port format. Example: 129.3.7.14:80, 129.3.7.15:80</p>
<p>cluster.proxyPort</p>	<p>80</p>	<p>[deprecated] The reverse proxy access port for the cluster. Use cluster.proxyIPList instead.</p>
<p>console.expiryErrInterval</p>	<p>10</p>	<p>Number of days before the cluster license expires to generate an error as a log message and a console indicator.</p>
<p>console.expiryWarnInterval</p>	<p>30</p>	<p>Number of days before the cluster license expires to generate a warning as a log message and a console indicator.</p>
<p>console.indexErrorLevel</p>	<p>90</p>	<p>Percentage, 0-100. How much index utilization generates an error as a log message and a console indicator.</p>
<p>console.indexWarningLevel</p>	<p>80</p>	<p>Percentage, 0-100. How much index utilization generates a warning as a log message and a console indicator.</p>
<p>console.messageExpirationSeconds SNMP: messageExpirationSeconds</p>	<p>1209600</p>	<p>In seconds; defaults to 2 weeks. How long until an error expires out of the error table.</p>
<p>console.port</p>	<p>90</p>	<p>Which port Swarm uses to listen for requests. All nodes in the same cluster must be set to the same port. When deploying Swarm into untrusted network environments, firewall this port so only administrators can access it.</p>

console.reportStyleUrl		The URL for the path to the stylesheet and image files for configuring Swarm console. Example: http://10.10.15.32/css/swarm-reports.css
console.spaceErrorLevel	10	Percentage, 0-100. How much cluster capacity remaining generates an error as a log message and a console indicator.
console.spaceWarnLevel	25	Percentage, 0-100. How much cluster capacity remaining generates a warning as a log message and a console indicator.
console.styleUrl		The URL for the path to the stylesheet and image files for configuring the Swarm console. Example: http://10.10.15.32/css/swarm.css
disk.atimeEnabled SNMP: accessedTimeEnabled	false	Whether to track the time of last access on GET requests, stored in the Castor-System-Accessed header and indexed as the search field 'accessed'. Increases load on the cluster and Elasticsearch.
disk.atimeGranularity SNMP: accessedTimeGranularity	86400	In seconds; defaults to 1 day. The window during which accessed time is not updated. Lowering the value affects GET performance.
disk.contextDeleteMarkerLifespan	31536000	In seconds; defaults to 1 year. How long a delete marker lives for a context (domain or bucket) object.
disk.deleteMarkerLifespan	1209600	In seconds; defaults to 2 weeks. How long the cluster remembers a deleted named object. Lower this value if applications create and delete objects so rapidly they cause available memory to decrease. To view the current amount of available memory on a node, expand Node Info to see the value of Index Utilization. A large number of objects may have been stored and may benefit from lowering this value if this value is high for a long period of time.
disk.obsoleteTimeout	1209600	In seconds; defaults to 2 weeks. The amount of time after which an unused volume is considered "stale" and does not recover, except with use of the 'k' modifier.
ec.conversionPercentage SNMP: ecConversionPercentage	0	Percentage, 1-100; 0 stops all conversion. Adjusts the rate at which the Health Processor consolidates multi-set erasure-coded objects each HP cycle. Lower to reduce cluster load; increase to convert a large number of eligible objects faster, at the cost of load on the cluster. Requires policy.ecEncoding to be specified.
ec.convertToPolicy SNMP: ecConvertToPolicy	false	When true, convert existing EC objects to the EC encoding specified by policy.
ec.convertVersionedObjects SNMP: ecConvertVersionedObjects	false	When true, Swarm performs lifepoint conversions and consolidations of multi-set erasure-coded versioned objects.
ec.maxManifests SNMP: ecMaxManifests	6	Range, 3-36. The maximum number of manifests written for an EC object. Usually p+1 are written for a k:p encoding. Do not set above 6 unless directed by Support.

ec.minParity SNMP: ecMinParity	-1	Range -1 or 1-4; default of -1 is $\max(\text{policyminreps} - 1, 1)$, where <code>policyminreps</code> is the min value in <code>policy.replicas</code> . The minimum number of parity segments the cluster requires. This is the lower limit on <code>p</code> for EC content protection, regardless of the parity value expressed on individual objects through query arguments or lifepoints.
ec.protectionLevel SNMP: ecProtectionLevel	node	Either 'node', 'subcluster', or 'volume'. At what level segments must be distributed for an EC write to succeed. Note: multiple segments are allowed per level if needed. 'node' (default) distributes segments across the cluster's physical/virtual machines. 'subcluster' requires <code>node.subcluster</code> to be defined across sets of nodes. $(k+p)/p$ nodes /subclusters for those levels are needed; at minimum, $k+p$ volumes are needed.
ec.segmentConsolidationFrequency SNMP: ecSegmentConsolidationFrequency	10	Percentage, 1-100, 0 to disable. How quickly the health processor consolidates object segments after ingest. Increase this value (such as to 25, to consolidate over 4 HP cycles) to make new content readable sooner by clients. For multipart uploads via S3 clients, 10 is recommended; for SwarmFS, 100 is recommended, with extra space allowances for trapped space. Consolidation changes the ETag (which affects If-Match requests) and Castor-System-Version headers, but Content-MD5 and Castor-System-CompositeMD5 headers are unchanged. Therefore, have clients use hash and last-modified date, rather than ETag, to find if an object has changed.
ec.segmentSize SNMP: ecSegmentSize	-1	In bytes; default of -1 implies 200 MB, with recommended minimum of 100 MB. The maximum size allowed for an EC segment before triggering another level of erasure coding. For mostly large (1+ GB) objects, increase to minimize the number of EC sets, which reduces index memory usage. Increase the size as needed per write request using the 'segmentsize' query argument.
feeds.retry SNMP: feedsRetryDelays	[30, 300, 1200]	In seconds. The progressive number of retry attempts by the plug-in, when blocked. Example: [60, 60, 60, 3600]
feeds.statsReportInterval	300	In seconds. How frequently to report statistics.
health.defragInterval SNMP: healthDefragInterval	3600	In seconds; defaults to 1 hour. How long to wait between attempts to defrag a volume during an HP cycle.
health.ecrSegmentDelay SNMP: healthFVRPushDelay	0.0	In seconds; defaults to 0.0. Tunes ECRs by defining the length of the forced delay after each segment is relocated. Change from default only as directed.
health.examDelay SNMP: healthExamDelay	0.18	In seconds; defaults to 0.18. Tunes the health processor by defining the length of the forced delay until the next HP exam, or removes the delay altogether (-1). Change from default only as directed.
health.fvrPushDelay SNMP: healthFVRPushDelay	0.3	In seconds; defaults to 0.3. Tunes FVRs by defining the length of the forced delay after each replica/bundle is pushed to another node. Change from default only as directed.

health.neonatalROWProtection	true	If the exam queue for newly written objects is close to overflow, enables Swarm to override the data protection scheme of transitioning to ROW (scsp.replicateOnWrite). All subsequent replicas are processed out of this queue.
health.offloadPauseInterval SNMP: healthOffloadPauseInterval	600	The delay between attempts to bulk offload to the cluster, in seconds.
health.parallelWriteTimeout SNMP: healthParallelWriteTimeout	2592000	In seconds; defaults to 1 month. When to time out an uncompleted multipart upload so Swarm can clean up the unused parts. 0 disables; do not disable if using SwarmFS.
health.persistentUnderreplicationAlertPercent SNMP: healthPersistentUnderreplicationAlertPercent	2	Percentage, 0-100; set 0 to disable. Creates an alert when this percentage (or more) of objects are persistently under-replicated.
health.recursiveDeleteDelay	604800	In seconds; defaults to 1 week. The length of the grace period before the health processor begins reclaiming the space for a deleted domain or bucket. During this grace period, the domain or bucket can be restored without losing any of the content. No grace period is granted if recursive=now is used.
health.relocationVolumeFillRate SNMP: hpRelocationVolumeFillRate	10	Percentage, 0-100. How much available space on new volumes may be filled for object relocation during one cluster health processor (HP) cycle, to prevent the HP on existing nodes from overwhelming a new, empty node.
health.replicationMulticastFrequency SNMP: repMulticastFrequency	1	Percentage, 0-100. The frequency, as an approximate percentage, UUIDs are multicast to verify replicas. Set this parameter to the same value for all nodes in the cluster.
health.replicationUnicastFrequency SNMP: repUnicastFrequency	100	Percentage, 0-100. The frequency, as an approximate percentage, a unit is forced to verify hints.
health.underreplicationAlertPercent SNMP: healthUnderreplicationAlertPercent	10	Percentage, 0-100; set 0 to disable. Generates an under-replication alert when the percentage of under-replicated objects exceeds this value.
health.underreplicationTolerance SNMP: healthUnderreplicationTolerance	100	Count. The number of under-replicated objects below which to suppress the alerts triggered by health.underreplicationAlertPercent.
index.optimize404 SNMP: overlayOptimize404	true	Enables the Optimize 404 feature in the overlay index, which returns 404 without multicast where possible.
index.ovMinNodes SNMP: overlayMinNodes	3	Count. The minimum number of cluster nodes needed to activate use of the overlay index.
index.overlayEnabled SNMP: overlayIndexEnabled	true	Enables the overlay index.
log.host SNMP: logHost		The IP address of the remote Syslog server. Logging must be used for production environments. Set to " to stop logging in test environments. Example: 10.10.33.12

log.level SNMP: logLevel	40	The log level, from most to least verbose, each including everything below it: 10, 20, 30, 40, 50, 0. 10 Debug (all information plus stack traces), 15 Audit (replication and object movement), 20 Info (informational, including non-errors), 30 Warn (user and application errors, plus SCSP 4xx/5xx codes), 40 Error (server hardware and software errors, plus abnormal conditions), 50 Critical (errors can result in data loss, such as disk I/O errors), 0 Disable logging.
log.obscureUUIDs SNMP: logObscureUUIDs	false	Whether to obscure UUIDs from displaying in INFO and higher level logs (does not affect AUDIT and lower levels). Set to True to abbreviate the UUID, if indicated by security requirements.
log.port SNMP: logPort	514	The port for the remote syslog host to use.
metrics.diskUtilizationCheckInterval	600	[deprecated] In seconds, from 15 seconds to 1 day; defaults to 10 minutes. How frequently to check disk utilization on the Elasticsearch cluster.
metrics.diskUtilizationThreshold	5	[deprecated] Percentage, 0-100. The minimum space available Elasticsearch disk space that, when reached, stop metrics from being indexed.
metrics.enableNodeExporter	true	Enabled by default. Set to FALSE to disable the node_exporter service, for the export of both node system metrics and Swarm metrics.
metrics.nodeExporterFrequency SNMP: metricsExporterFrequency	0	In seconds, from 1 minute to 1 hour; How frequently to refresh Swarm-specific metrics via the node exporter. 0 disables export of this data.
metrics.period SNMP: metricsPeriod	900	[deprecated] In seconds, from 15 seconds to 1 day; defaults to 15 minutes. How frequently to capture metrics-related statistics.
metrics.port SNMP: metricsTargetPort	9200	[deprecated] The port on the Elasticsearch server where metrics-related statistics are captured.
metrics.target SNMP: metricsTargetHost		[deprecated] One or more servers in the Elasticsearch cluster (fully qualified domain names or IP addresses) where metrics-related statistics are captured. Use spaces or commas to separate multiple values. To disable statistics collection, leave blank. Examples: http://es1.company.com , http://es2.company.com 10.12.14.14
network.dnsDomain		Optional. The domain name(s) searched for host name resolution when using static IP assignment. Ignored unless network.ipAddress is set. Use in conjunction with network.dnsServers. Examples: example.com hq.example.com dr.example.com

network.dnsServers		Optional. The servers used for host name resolution when using static IP assignment. Ignored unless network.ipAddress is set. Use in conjunction with network.dnsDomain. Examples: 8.8.8.8 1.1.1.1 8.8.4.4
network.icmpAcceptRedirects	true	Determines if the node accepts routing information from ICMP redirect responses.
network.igmpTimeout SNMP: networkIGMPTimeout	0	In seconds; defaults to 0 (disabled). The IGMP querier timeout, which is the frequency IGMP queries are sent on the network.
network.igmpVersion	2	Range, 1-3. The IGMP (Internet Group Management Protocol) version the Linux kernel uses for host membership queries.
network.mtu	0	In bytes. Sets the maximum transmission unit (MTU) Swarm accepts. Set to a higher value to use jumbo frames. Verify the node's network interfaces and all other network hardware support the selected MTU before the default value is changed; otherwise, the nodes might not be able to replicate objects or communicate. Set to 0 to use value from DHCP or else 1500.
policy.eCEncoding SNMP: policyECEncoding	unspecified anchored	The cluster-wide setting for the EC (erasure coding) encoding policy. Valid values: unspecified, disabled, k:p (a tuple such as 5:2 specifies the data (k) and parity (p) encoding to use). Add 'anchored' to set this cluster-wide; remove it to allow domains and buckets to have custom encodings. Examples: 5:2 6:3 anchored
policy.eCMinStreamSize SNMP: policyECMinStreamSize	1Mb anchored	In integer units of megabytes (MB) or gigabytes (GB); must be 1MB or greater. The size that triggers an object to be erasure-coded, if specified (by eCEncoding, lifepoint, query argument) and allowed by policy. Below this threshold, objects are replicated unless they are multipart or chunked writes. Add 'anchored' to set this cluster-wide; remove it to allow domains and buckets to have custom values. Examples: 100Mb 1GB anchored
policy.lifecycle SNMP: policyLifecycle	disabled	The cluster-wide setting for bucket lifecycle policies. If enabled, bucket lifecycle policies are evaluated. Examples: disabled enabled
policy.replicas SNMP: policyReplicas	min:2 max:16 default: 2 anchored	The minimum, maximum, and default number of replicas allowed for objects in this cluster. Can differ from the policy in a replicated target cluster. Examples: min:2 max:16 default:3 min:3 max:10 default:3

<p>policy.versioning SNMP: policyVersioning</p>	<p>disallowed</p>	<p>Specifies whether versioning is allowed to be enabled on contexts (domains and buckets) within the cluster. Valid states: disallowed, suspended, allowed. This policy overrides context-level policies. Disallowed removes historical versions, if any. Suspended stops creation of new versions but retains version history. Examples: allowed disallowed suspended</p>
<p>power.savingMode SNMP: powerSavingMode</p>	<p>true</p>	<p>Enables Power Saving mode, which allows the system to sleep or power cap. Set to False to disable Power Saving mode.</p>
<p>power.sleepAfter SNMP: sleepAfter</p>	<p>7200</p>	<p>In seconds, 60 or greater; defaults to 2 hours. In Power Saving mode, how long a node is inactive before it becomes idle.</p>
<p>power.wakeAfter SNMP: wakeAfter</p>	<p>28800</p>	<p>In seconds; defaults to 8 hours. In Power Saving mode, how long a node is idle before it becomes active again.</p>
<p>recovery.completedRecoveryExpiration SNMP: completedRecoveryExpiration</p>	<p>2592000</p>	<p>In seconds; defaults to 30 days. How long to remember completed recoveries.</p>
<p>recovery.suspend SNMP: volumeRecoverySuspend</p>	<p>false</p>	<p>Defaults to False, which allows normal volume recovery and recovery behavior. Set to True to disable all recovery behavior. All nodes in the cluster must be set to the same value.</p>
<p>recovery.suspendedVolumes SNMP: castorAddVolumeRecoverySuspend, castorRemoveVolumeRecoverySuspend</p>	<p>[]</p>	<p>The comma-separated list of 32-character volume IDs of the volumes for which recovery is suspended. Example: ['d315ca82bae4b4a0d24fd90904216554', '2195a057c205bd58e05f5835d4b9f21e']</p>
<p>recovery.volMaintenanceInterval SNMP: volMaintenanceInterval</p>	<p>10800</p>	<p>In seconds; defaults to 3 hours. How long the cluster waits after a node has been rebooted or shut down before considering the node and its volumes missing for recovery and replication purposes. This time does not include the time to mount the volumes. This maintenance window allows administrators to perform regular, scheduled tasks on a node without creating over-replication in the cluster. Node shutdowns or failures not initiated by an administrator are considered immediately missing.</p>
<p>scsp.allowPutCreate SNMP: allowPutCreate</p>	<p>false</p>	<p>When true, PUTs can be used to create new named objects. Conditional headers still apply. With this option enabled, the putcreate query argument does not need to be added.</p>
<p>scsp.autoContentMD5Computation SNMP: autoContentMD5Computation</p>	<p>false</p>	<p>When true, Swarm computes and stores the Content-MD5 value on every applicable write.</p>
<p>scsp.autoRecursiveDelete SNMP: autoRecursiveDelete</p>	<p>true</p>	<p>When true, all context deletes (deletes of domains and buckets) are treated as recursive, which prevents orphaned content. The recursive query argument does not need to be add with this option enabled. Use the recursive=now argument to force immediate reclamation of space.</p>
<p>scsp.clientPoolTimeout SNMP: scspClientPoolTimeout</p>	<p>120</p>	<p>In seconds. How long until pooled SCSP connections expire.</p>

scsp.defaultContextReplicas SNMP: scspDefaultContextReplicas	-1	Defaults to -1, which uses the value of scsp.maxContextReplicas. Sets the default number of replicas for a POST/PUT on a context (domain or bucket) object if the number is not specified by the current lifepoint or the request.
scsp.defaultFeedSendTimeout	30.0	The timeout on a feed SEND request, if the timeout=true query argument is provided.
scsp.defaultROWAction SNMP: scspDefaultROWAction	immediate	The default Replicate On Write (ROW) action when scsp.replicateOnWrite is enabled. Valid options are 'immediate', 'full', or an integer between 2 and 5 (inclusive).
scsp.domainHeaders	['X-Forwarded-Host', 'Host']	A comma-separated list of headers that specifies the search order in which to find the host of an SCSP request. RFC 7230 5.4 requires a Host header with every SCSP request to support web servers or server farms hosting multiple domains. A client may use an HTTP proxy that modifies the Host header, but the Swarm domain name matches the original Host header. An HTTP proxy copies the original Host header into another header, typically X-Forwarded-Host. Examples: ['X-Forwarded-Host', 'Host', 'X-ProxyForward-Host'] ['Host']
scsp.enableVolumeRedirects SNMP: enableVolumeRedirects	false	Whether to allow redirects to SCSP heads on volume processes, for faster GET requests. For use with Gateway only, and best for sites with smaller objects.
scsp.falseStartTimeout	240	In seconds, 0 to disable; defaults to 4 minutes. How long to wait to receive the first byte before timing out and disconnecting.
scsp.filterResponseBlacklist SNMP: filterResponseBlacklist	[]	Which headers to remove from HTTP responses. List is comma-separated and case-insensitive: ['Castor-System-Path', 'Castor-System-Owner']
scsp.filterResponseHeaders SNMP: filterResponseHeaders	none	Swarm filters response headers according to the given method. Allowed values: 'none', 'blacklist', 'whitelist'.
scsp.filterResponseWhitelist SNMP: filterResponseWhitelist	[]	Which headers to retain in HTTP responses, removing all others. List is comma-separated and case-insensitive: ['Etag', 'Last-Modified']
scsp.idleDisconnectTimeout	14400	In seconds, 0 to disable; defaults to 4 hours. How long to wait after receiving the last byte before timing out and disconnecting.
scsp.keepAliveInterval SNMP: keepAliveInterval	15	How many seconds to wait before sending successive chunked keep-alive bytes after a 202 Accepted response.
scsp.maxContextReplicas SNMP: maxcontextreplicas	16	Count. Sets the maximum number of replicas in this cluster for a context (domain or bucket) object.
scsp.maxReadTime	10800	SCSP read time limit in seconds; defaults to 3 hours. SCSP GET requests running longer than this value are prematurely closed.
scsp.maxWriteTime SNMP: scspMaxWriteTime	10800	SCSP write time limit in seconds; defaults to 3 hours. SCSP write requests running longer than this value are prematurely closed.

scsp.port SNMP: scspport	80	Port number; defaults to 80. The port used by client applications to access cluster nodes with HTTP requests. This setting must be the same on all nodes in the same cluster.
scsp.replicateOnWrite SNMP: autoRepOnWrite	true	Enabled by default. Improves content integrity by requiring a replica be written for the POST, PUT, COPY, or APPEND request to succeed. Set to False to have the health processor manage creation of replicas after the write.
scsp.requireExplicitContextCreate SNMP: requireExplicitContextCreate	false	When true, Swarm requires creation of a context (domain or bucket) to include the 'Content-type: application/castorcontext' header. Enable the option to protect against content being erroneously written as context objects, which hurts performance.
scsp.validateOnRead SNMP: scspValidateOnRead	false	Disabled by default. Enable to force Swarm to validate the object's contents before returning successful read responses to client requests. Although validation can be specified on a per-read basis, this setting forces all reads to use validation. During the read from the disk, the content hash is computed. If the hash is wrong, indicating logical disk corruption, the socket is closed before the last block is transmitted, forcing an error to the client. Note: using this option creates additional CPU load on the node.
search.caseInsensitive	false	Whether metadata fields should support case-insensitive searching. If true, then all custom metadata is indexed to support only case-insensitive searching.
search.enableCustomMetadataTyping SNMP: enableCustomMetadataTyping	true	Whether to publish custom metadata typing information to Elasticsearch.
search.enableDelimiterPaths SNMP: enableDelimiterPaths	false	Whether to publish name delimiter path information to Elasticsearch.
search.numberOfShards SNMP: searchNumberOfShards	5	The number of shards to use when creating new Elasticsearch search indexes.
search.pathDelimiter	/	Which character to use for parsing directory paths from object names, such as '2018/Q4/snapshot.pdf'. Defaults to forward slash: /
security.administrators SNMP: addModifyAdministrator, removeAdministrator	{'admin': 'ourpwdofchoicehere'}	One or more username:password pairs. Sets credentials for who can administer the cluster via the Swarm UI. If the value includes the snmp username, remove it from here and update snmp.rwCommunity with its password. Example: {'admin': 'adminpassword', 'admin2': 'adminpassword2'}
security.noauth	true	[deprecated] To enable native Swarm authorization, set to False.
security.operators	{}	One or more username:password pairs. Sets credentials for who can view the Swarm UI. If the value includes an snmp username, it is ignored; remove it from here and update snmp.roCommunity with its password. Example: {'operator': 'operatorpassword', 'operator2': 'operatorpassword2'}

security.secureLogging SNMP: secureLogging	false	Enable to prevent logging of the details of a client request. This option results in short, secure log messages.
snmp.getnextskips	['35', '36.20', '36.21', '36.22', '36.23', '36.25', '37.11.8', '38', '41', '55', '57', '58', '61', '63', '64', '65', '66', '68', '69']	List of OIDs to be skipped on output. To protect cluster performance, this setting causes the snmpwalk of the entire CASTOR MIB to skip several large, detailed tables in SNMP groups. The default list of OIDs causes a top-level snmpwalk to skip the groups or tables under clusterConfig, responseHistogramTable, hp, clusterdata, indexer, configVariableTable, castorFeeds, feedVolTable, performance, and recoveryTable. Add or remove OIDs to control which sections of the MIB are returned by an snmpwalk. Enter values as strings in numeric form, relative to the Castor OID, .1.3.6.1.4.1.24659.1. Example: ['35', '37.11.8', '38', '41', '55', '57', '58', '61', '63', '64', '65', '66', '68', '69']
snmp.roCommunity	public	Password for the SNMP read-only community. If security.operators includes the snmp username, remove it and update the password here.
snmp.rwCommunity	ourpwdofchoicehere	Password for the SNMP read-write community. If security.administrators includes the snmp username, remove it and update the password here.
snmp.timeout SNMP: snmpTimeout	5	In seconds, 1-60. The snmpget, snmpset, and snmpwalk timeout for Swarm and Watchdog.
startup.certificates		Public certificates to add to cert bundle.

Chassis (Node) Settings

Name (SNMP)	Default	Description & Examples
cache.expirationTime	600	In seconds; defaults to 10 minutes. Set 0 to disable. How long to hold an object after its last access.
cache.maxCacheableSize	1048576	In bytes, defaulting to 1 MB. The largest object able to be stored in the content cache. If increased to greater than 5 MB, then scsp.readBufferAllowance must be increased to the same value.
cache.percentage	10	Percentage, 0-100; set 0 to disable. How much I/O buffer memory to reserve for the content cache, which improves access to active content by storing it in geographically proximate locations. The reserve is reported when the node starts up: 'MAIN ANNOUNCE: Memory allocation at startup.' For best performance, especially with writing named objects, do not disable the content cache unless directed by Support.
cache.realmStaleTimeout	600	In seconds, 60 or higher. How long before the security user list cache for domains is cleared. Lower this value if user lists update frequently.
chassis.name		The user-defined chassis name.
cip.histogramInterval	0.01	In seconds. The histogram bucket bin size.
cip.queryMinimumTimeout	0.0	In seconds. The minimum CIP query session time.

cip.queryTimeout	0.03	In seconds. How long after booting the cluster initially waits for node replication bids. Once the cluster is running, bid wait times are calculated dynamically based on response times. For clusters with network latency, the initial wait time may need to be increased until the cluster can correctly calibrate.
cip.readBufferSize	1048576	In bytes. The size of the multicast UDP socket read buffer. This value is capped the Linux <code>/proc/sys/net/core/rmem_max</code> value, which may be set via <code>sysctl.coreRMemMax</code> .
disk.defragBufferBytes	0	Size in bytes of the per-disk buffer allocated for bulk defragmentation operations. Disable bulk mode by setting to 0.
disk.defragUntilPercentage	0.8	Ratio, 0.0-1.0. The portion of known unused space that, when untrapped, stops the disk defrag process.
disk.enableMultipath	false	[deprecated] Whether to enable support for Device Mapper Multipathing (DM-Multipath). Multipath support was dropped in Swarm 10.0.
disk.encryptNewVolumes	false	Whether to encrypt new Swarm volumes. Enabling <code>encryptNewVolumes</code> means any newly-formatted Swarm volume are encrypted
disk.encryptionCipher	aes-xts-plain64	The encryption cipher to be used when setting encryption for new Swarm volumes. Supported values are <code>aes-xts-plain64</code> and <code>aes-cbc-essiv</code>
disk.encryptionHash	sha512	The encryption hash algorithm to be used when setting encryption for new Swarm volumes. Supported values are <code>sha256</code> and <code>sha512</code> .
disk.encryptionIterationTime	5000	In seconds. The maximum amount of time to be spent while iterating to generate an internal LUKS key from a Swarm encryption key, which is used when setting encryption for new Swarm volumes.
disk.encryptionKeyPrimary		The mnemonic name of the encryption key to use for encrypting new Swarm volumes. Do not use quotes. For this key to be used, <code>disk.encryptNewVolumes</code> must be set to <code>True</code> . Example: <code>cluster_key_5_15_2016</code>
disk.encryptionKeySize	512	The size of the internal LUKS key to be used when setting encryption for new Swarm volumes. Supported values are 128, 256, and 512.
disk.encryptionKeys	{}	A comma-separated list of mnemonic name and encryption key pairs, used for accessing encrypted Swarm volumes. Example: {'cluster_key_5_15_2016': 'a24f8ec391ab3341', 'cluster_key_5_12_2015': 'de3498245ce8bf89'}
disk.encryptionType	luks	The encrypted volume format type used when formatting new volumes. Supported values are: <code>'luks'</code> , <code>'luks1'</code> , <code>'luks2'</code> .
disk.ioErrorToRetire	2	Count. How many consecutive I/O errors (no more than <code>disk.ioErrorWindow</code> seconds between each error) that forces a volume to retire.
disk.ioErrorTolerance	200	Count. How many I/O errors are tolerated, past which the volume is taken offline immediately. Swarm marks the volume as Unavailable and initiates both the volume recovery process (FVR) and the erasure coding recovery process (ECR) to relocate all objects on the volume.

disk.ioErrorWindow	172800	In seconds; defaults to 2 days. The length of time after which an I/O error is forgotten, if no other errors followed and the volume's state is OK. Works with disk.ioErrorToRetire to control when volumes are retired. The default value means the volume is retired if more than one error occurs within 2 days.
disk.minGB	64	Minimum capacity in GB a storage device needs to be eligible for automatic storage volume assignment with volumes = all. Set to 0 to include all disk devices.
disk.smudgesToRetire	4	How many soft errors (smudges) over the life of a volume triggers Swarm to retire the volume. A soft error occurs when the health processor does not get the expected data when validating the object but the disk gave no explicit I/O (hard) error. Set to 0 to disable the automatic retire.
disk.standbyTimeout	360	In seconds. How long until an idle disk spins down automatically.
disk.trappedToTotalPercentage	0.0001	Ratio, 0.0-0.01. The portion, of trapped space to total space, below which, stops the defrag process (0.0 for no limit).
disk.volumes SNMP: vols		Required. Specifies the volume storage devices for Swarm to use. Valid entries: all, or a space-separated list of Linux volume identifiers, such as /dev/sda, /dev/sdb. all (recommended) is required for hot plugging and lets Swarm to use all volumes greater than disk.minGB. All volumes are stale and cannot be used unless a volume remount is forced by adding the :k (keep) policy option modifier if a node is shut down longer than disk.obsoleteTimeout. Add a modifier with units: vols1:100m vols2:250g to specify the size. Examples: all /dev/sda /dev/sdb
ec. inProgressConsolidationTimeout	86400	Time in seconds, 0 to disable. An 'in progress' multipart PATCH complete cannot be consolidated before this timeout.
feeds.maxMem	100000	In bytes. The maximum memory allowed per feed, for queue management.
health.startDelay SNMP: hpStartDelay	900	In seconds; defaults to 900 seconds (15 minutes). How long after a node starts up to begin Health Processor checking and recovery processes. This option creates a grace period for the remaining nodes to stabilize in the cluster, which is useful in situations in which an entire cluster must be shut down and restarted.
license.url	<Swarm default 2T license>	The location and name of the Swarm license file, caringo/license.txt. Can be a pathname or a URL. The default location must be kept to use the default 2 TB license. Example: http://10.10.15.32/config/swarm-license.txt
mdns.readBufferSize	1048576	In bytes. The size of the read buffer for the multicast UDP socket.
network.gateway SNMP: gateway		Optional. The default gateway IP address in the subnet. Ignored unless network.ipAddress is set. Example: 10.10.12.253
network.ipAddress SNMP: ipaddress		The single static IP address for a node to use, or blank to use DHCP. Example: 10.10.12.1

network.iptablesFileUrl		Optional. Location (URL) of Linux firewall rules to apply. When specified, Swarm transmits the rules without validation to the 'iptables-restore' command before starting the storage node processes. Example: http://10.10.15.32/config/swarm-iptables
network.netmask SNMP: netmask		Optional. Sets the IP network mask for a node. Ignored unless network.ipAddress is set. Examples: 255.255.255.0 255.255.0.0
network.timeSource SNMP: timeSource		Recommended. List of one or more NTP servers by IP address or by name if network.dnsServers is set. One usable NTP server needs to be configured for the storage node to start. A list from *.pool.ntp.org is generated if a value is not assigned here. Examples: 10.20.30.55 10.20.30.65 0.be.pool.ntp.org 1.be.pool.ntp.org
node.archiveMode SNMP: archiveMode	false	Disabled by default, which is the normal operating state. Set to TRUE to change the node to archive mode, where it remains idle in low-power mode without participating in cluster activity until its capacity is needed. This setting is useful for proactively provisioning new nodes into the cluster before they are needed.
node.subcluster SNMP: subcluster	default	Specifies the name of the subcluster to which the chassis belongs. Names can have no more than 16 characters and no special characters, such as quotes and hyphens. Example: subcluster1
shutdown.gracePeriod	120	In seconds; defaults to 2 minutes. How long to allow ongoing SCSP requests to complete during shutdown.
snmp.enabled	true	Master switch to enable or disable the SNMP daemon
snmp.sysContact	Unspecified	The value for the SNMP system contact, SNMPv2-MIB::sysContact. Must be a valid email address in 7-bit USASCII in one of these forms: Name <email@domain> First Last <email@domain> Example: mailto:admin@company.com
snmp.sysLocation	Unspecified	The value for the SNMP system location, SNMPv2-MIB::sysLocation. Example: rack3
snmp.sysName	Unspecified	The value for the SNMP system name, SNMPv2-MIB::sysName. Example: Joe Administrator

Replaced Settings

The following table lists obsolete settings and corresponding new, qualified names. Update the configuration to use the new settings:

1. Update `node.cfg/cluster.cfg` files.
2. Update persisted settings via SNMP. See [Persisted Settings \(SNMP\)](#).

Warning

Incorrectly renaming a setting can prevent the node or cluster from booting.

The following table lists settings alphabetically by the deprecated names.

Old Setting	New Setting
autoRepOnWrite	scsp.replicateOnWrite
autoValidateRead	scsp.validateOnRead
chassis.processes	<i>No longer needed, v10.0</i>
cipTTL	cip.ttl
cluster	cluster.name
cluster.proxyPort cluster.proxyIPAdress	cluster.proxyIPList
cluster.settingsUuid	<i>No longer needed</i>
consolePort	console.port
consoleReportStyleURL	console.reportStyleUrl
consoleStyleURL	console.styleUrl
crier.deadVolumeWall	<i>Removed, v11.1</i>
defreps	policy.replicas
disk.automaticFormat	<i>No longer supported</i>
ec.encoding	policy.ecEncoding
ec.minStreamSize	policy.ecMinStreamSize
ec.subclusterLossTolerance	<i>No longer needed, v10.0</i>
hpStartDelay	health.startDelay
ipaddress	network.ipAddress

licenseFileURL	license.url
loghost	log.host
loglevel	log.level
logport	log.port
maxreps minreps	policy.replicas
networkMTU	network.mtu
realmCacheStaleTimeout	cache.realmStaleTimeout
repMulticastFrequency	health.replicationMulticastFrequency
repPriority	health.replicationPriority
repThreshold	bidding.relocationThreshold
scsp.minReplicas scsp.maxReplicas scsp.defaultReplicas	policy.replicas <i>required parameters: min, max, default</i>
scspport	scsp.port
security.noauth	<i>No longer supported</i>
snmpSysContact	snmp.sysContact
snmpSysLocation	snmp.sysLocation
snmpSysName	snmp.sysName
spaceErrLevel	console.spaceErrorLevel
volMinimumGB	disk.minGB
volPluginURL	disk.volumeIdentifyUrl
vols	disk.volumes
volStandbyTimeout	disk.standbyTimeout
volumeRecoverySuspend	recovery.suspend

Alphabetical Settings List

Here is an alphabetical listing of Swarm Storage settings with section qualifiers removed.



Tip

Use the *complete* qualified form for best search results: `section.setting` when searching on setting names throughout this documentation.

Setting Name	Section + Setting Name (use for search)
administrators	security.administrators
allowPutCreate	scsp.allowPutCreate
archiveMode	node.archiveMode
atimeEnabled	disk.atimeEnabled
atimeGranularity	disk.atimeGranularity
autoContentMD5Computation	scsp.autoContentMD5Computation
autoRecursiveDelete	scsp.autoRecursiveDelete
caseInsensitive	search.caseInsensitive
certificates	startup.certificates
clientPoolTimeout	scsp.clientPoolTimeout
completedRecoveryExpiration	recovery.completedRecoveryExpiration
contextDeleteMarkerLifespan	disk.contextDeleteMarkerLifespan
conversionPercentage	ec.conversionPercentage
convertToPolicy	ec.convertToPolicy
convertVersionedObjects	ec.convertVersionedObjects
defaultContextReplicas	scsp.defaultContextReplicas
defaultFeedSendTimeout	scsp.defaultFeedSendTimeout
defaultROWAction	scsp.defaultROWAction
defragInterval	health.defragInterval
defragUntilPercentage	disk.defragUntilPercentage

deleteMarkerLifespan	disk.deleteMarkerLifespan
diskUtilizationCheckInterval	metrics.diskUtilizationCheckInterval
diskUtilizationThreshold	metrics.diskUtilizationThreshold
dnsDomain	network.dnsDomain
dnsServers	network.dnsServers
domainHeaders	scsp.domainHeaders
eCEncoding	policy.eCEncoding
eCMinStreamSize	policy.eCMinStreamSize
enableCustomMetadataTyping	search.enableCustomMetadataTyping
enabled	snmp.enabled
enableDelimiterPaths	search.enableDelimiterPaths
enableNodeExporter	metrics.enableNodeExporter
enableVolumeRedirects	scsp.enableVolumeRedirects
encryptionCipher	disk.encryptionCipher
encryptionHash	disk.encryptionHash
encryptionIterationTime	disk.encryptionIterationTime
encryptionKeyPrimary	disk.encryptionKeyPrimary
encryptionKeys	disk.encryptionKeys
encryptionKeySize	disk.encryptionKeySize
encryptionType	disk.encryptionType
encryptNewVolumes	disk.encryptNewVolumes
enforceTenancy	cluster.enforceTenancy
examDelay	health.examDelay
expirationTime	cache.expirationTime
expiryErrInterval	console.expiryErrInterval
expiryWarnInterval	console.expiryWarnInterval
falseStartTimeout	scsp.falseStartTimeout
filterResponseBlacklist	scsp.filterResponseBlacklist
filterResponseHeaders	scsp.filterResponseHeaders

filterResponseWhitelist	scsp.filterResponseWhitelist
gateway	network.gateway
getnextskips	snmp.getnextskips
gracePeriod	shutdown.gracePeriod
group	cip.group
host	log.host
icmpAcceptRedirects	network.icmpAcceptRedirects
idleDisconnectTimeout	scsp.idleDisconnectTimeout
igmpTimeout	network.igmpTimeout
igmpVersion	network.igmpVersion
indexErrorLevel	console.indexErrorLevel
indexWarningLevel	console.indexWarningLevel
inProgressConsolidationTimeout	ec.inProgressConsolidationTimeout
ioErrorTolerance	disk.ioErrorTolerance
ioErrorToRetire	disk.ioErrorToRetire
ioErrorWindow	disk.ioErrorWindow
ipAddress	network.ipAddress
iptablesFileUrl	network.iptablesFileUrl
keepAliveInterval	scsp.keepAliveInterval
level	log.level
lifecycle	policy.lifecycle
maxCacheableSize	cache.maxCacheableSize
maxContextReplicas	scsp.maxContextReplicas
maxManifests	ec.maxManifests
maxMem	feeds.maxMem
maxReadTime	scsp.maxReadTime
maxWriteTime	scsp.maxWriteTime
messageExpirationSeconds	console.messageExpirationSeconds
minGB	disk.minGB

minParity	ec.minParity
mtu	network.mtu
name	chassis.name
name	cluster.name
neonatalROWProtection	health.neonatalROWProtection
netmask	network.netmask
noauth	security.noauth
nodeExporterFrequency	metrics.nodeExporterFrequency
numberOfShards	search.numberOfShards
obscureUUIDs	log.obscureUUIDs
obsoleteTimeout	disk.obsoleteTimeout
offloadPauseInterval	health.offloadPauseInterval
operators	security.operators
optimize404	index.optimize404
overlayEnabled	index.overlayEnabled
ovMinNodes	index.ovMinNodes
parallelWriteTimeout	health.parallelWriteTimeout
pathDelimiter	search.pathDelimiter
percentage	cache.percentage
period	metrics.period
persistentUnderreplicationAlertPercent	health.persistentUnderreplicationAlertPercent
port	console.port
port	log.port
port	metrics.port
port	scsp.port
protectionLevel	ec.protectionLevel
proxyIPAddress	cluster.proxyIPAddress
proxyIPList	cluster.proxyIPList
proxyPort	cluster.proxyPort

queryRetryMultiplier	cip.queryRetryMultiplier
queryTimeout	cip.queryTimeout
readBufferSize	cip.readBufferSize
readBufferSize	mdns.readBufferSize
realmStaleTimeout	cache.realmStaleTimeout
recursiveDeleteDelay	health.recursiveDeleteDelay
relocationThreshold	bidding.relocationThreshold
relocationVolumeFillRate	health.relocationVolumeFillRate
replicas	policy.replicas
replicateOnWrite	scsp.replicateOnWrite
replicationMulticastFrequency	health.replicationMulticastFrequency
replicationUnicastFrequency	health.replicationUnicastFrequency
reportStyleUrl	console.reportStyleUrl
requireExplicitContextCreate	scsp.requireExplicitContextCreate
retry	feeds.retry
roCommunity	snmp.roCommunity
rwCommunity	snmp.rwCommunity
savingMode	power.savingMode
secureLogging	security.secureLogging
segmentConsolidationFrequency	ec.segmentConsolidationFrequency
segmentSize	ec.segmentSize
sleepAfter	power.sleepAfter
smudgesToRetire	disk.smudgesToRetire
spaceErrorLevel	console.spaceErrorLevel
spaceWarnLevel	console.spaceWarnLevel
standbyTimeout	disk.standbyTimeout
startDelay	health.startDelay
statsReportInterval	feeds.statsReportInterval
styleUrl	console.styleUrl

subcluster	node.subcluster
suspend	recovery.suspend
suspendedVolumes	recovery.suspendedVolumes
sysContact	snmp.sysContact
sysLocation	snmp.sysLocation
sysName	snmp.sysName
target	metrics.target
timeout	snmp.timeout
timeSource	network.timeSource
trappedToTotalPercentage	disk.trappedToTotalPercentage
ttl	cip.ttl
underreplicationAlertPercent	health.underreplicationAlertPercent
underreplicationTolerance	health.underreplicationTolerance
url	license.url
validateOnRead	scsp.validateOnRead
versioning	policy.versioning
volMaintenanceInterval	recovery.volMaintenanceInterval
volumes	disk.volumes
wakeAfter	power.wakeAfter

Persisted Settings (SNMP)

A subset of Swarm configuration settings are *persisted* settings, which are stored in a Settings object in your cluster if you have any domains or have ever changed settings in the Swarm UI (or legacy Admin Console). These special settings persist across reboots, regardless of how you may have updated your configuration (node.cfg/cluster.cfg) files.

Important

The Settings object persists and overrides the configuration files, storing both settings and passwords for the cluster. See [Swarm Passwords](#).

Best practice – Always change settings via the [Swarm UI](#), rather than through the configuration files. There are several benefits to this practice:

- **No reboot required.** You do not need a full reboot for the configuration change to take effect.
- **Updates persisted settings.** The changes are stored directly in the Settings object.
- **Only update in one place.** You only need to update persisted settings on *one* node: Swarm propagates the changes to all other Settings objects in the cluster.

SNMP version

Swarm supports SNMP version 2 only.

Here is an example SNMP set command that changes the *string* that is the policy for cluster-wide versioning:

```
snmpset -m +CARINGO-CASTOR-MIB -v2c -M +/usr/share/snmp/mib2c-data -c{password} -Oqs {node} policyVersioning s "allowed"
```

Here is an example that changes an *integer* value:

```
snmpset -m +CARINGO-CASTOR-MIB -v2c -M +/usr/share/snmp/mib2c-data -c{password} -Oqs {node} healthExamDelay i 30
```

Note

{node} is the IP address of any Swarm storage node. **{password}** is the SNMP read-write community as specified via the snmp.rwCommunity cluster setting.

Listed below are the special Swarm settings with SNMP names, as well as whether they are persisted and writable. All are settable via the [Swarm UI](#)

Name	Writable	Scope	Persisted	SNMP Name
bidding.relocationThreshold	Yes	cluster	Yes	relocationThreshold
cip.group		cluster		group
cip.queryRetryMultiplier	Yes	cluster	Yes	queryRetryMultiplier

cluster.enforceTenancy	Yes	cluster		enforceTenancy
cluster.name		cluster		cluster
console.messageExpirationSeconds	Yes	cluster	Yes	messageExpirationSeconds
disk.atimeEnabled	Yes	cluster	Yes	accessedTimeEnabled
disk.atimeGranularity	Yes	cluster	Yes	accessedTimeGranularity
disk.volumes		node		vols
ec.conversionPercentage	Yes	cluster	Yes	ecConversionPercentage
ec.convertToPolicy	Yes	cluster	Yes	ecConvertToPolicy
ec.convertVersionedObjects	Yes	cluster	Yes	ecConvertVersionedObjects
ec.protectionLevel	Yes	cluster	Yes	ecProtectionLevel
ec.segmentConsolidationFrequency	Yes	cluster	Yes	ecSegmentConsolidationFrequency
health.defragInterval	Yes	cluster	Yes	healthDefragInterval
health.examinationDelay	Yes	cluster	Yes	healthExamDelay
health.offloadPauseInterval	Yes	cluster	Yes	healthOffloadPauseInterval
health.relocationVolumeFillRate	Yes	cluster	Yes	hpRelocationVolumeFillRate
health.replicationMulticastFrequency	Yes	cluster	Yes	repMulticastFrequency
health.replicationUnicastFrequency	Yes	cluster	Yes	repUnicastFrequency
health.startDelay		node		hpStartDelay
index.optimize404	Yes	cluster	Yes	overlayOptimize404
index.overlayEnabled	Yes	cluster	Yes	overlayIndexEnabled
index.ovMinNodes	Yes	cluster	Yes	overlayMinNodes
log.host	Yes	cluster	Yes	logHost
log.level	Yes	cluster	Yes	logLevel
log.port	Yes	cluster	Yes	logPort
metrics.nodeExporterFrequency	Yes	cluster	Yes	metricsExporterFrequency
metrics.period	Yes	cluster	Yes	metricsPeriod
metrics.port	Yes	cluster	Yes	metricsTargetPort
metrics.target	Yes	cluster	Yes	metricsTargetHost
network.gateway		node		gateway
network.igmpTimeout	Yes	cluster	Yes	networkIGMPTimeout
network.ipAddress		node		ipaddress
network.netmask		node		netmask
network.timeSource		node		timeSource
node.archiveMode	Yes	node		archiveMode

node.subcluster	Yes	node		subcluster
policy.eCEncoding	Yes	cluster	Yes	policyECEncoding
policy.eCMinStreamSize	Yes	cluster	Yes	policyECMinStreamSize
policy.lifecycle	Yes	cluster	Yes	policyLifecycle
policy.replicas	Yes	cluster	Yes	policyReplicas
policy.versioning	Yes	cluster	Yes	policyVersioning
power.savingMode	Yes	cluster	Yes	powerSavingMode
power.sleepAfter	Yes	cluster	Yes	sleepAfter
power.wakeAfter	Yes	cluster	Yes	wakeAfter
recovery.completedRecoveryExpiration	Yes	cluster	Yes	completedRecoveryExpiration
recovery.suspend	Yes	cluster	Yes	volumeRecoverySuspend
recovery.suspendedVolumes	Yes	cluster	Yes	castorAddVolumeRecoverySuspend, castorRemoveVolumeRecoverySuspend
recovery.volMaintenanceInterval	Yes	cluster	Yes	volMaintenanceInterval
scsp.allowPutCreate	Yes	cluster	Yes	allowPutCreate
scsp.autoContentMD5Computation	Yes	cluster	Yes	autoContentMD5Computation
scsp.autoRecursiveDelete	Yes	cluster	Yes	autoRecursiveDelete
scsp.filterResponseBlacklist	Yes	cluster	Yes	filterResponseBlacklist
scsp.filterResponseHeaders	Yes	cluster	Yes	filterResponseHeaders
scsp.filterResponseWhitelist	Yes	cluster	Yes	filterResponseWhitelist
scsp.keepAliveInterval	Yes	cluster	Yes	keepAliveInterval
scsp.maxContextReplicas	Yes	cluster	Yes	maxcontextreplicas
scsp.port		cluster		scspport
scsp.replicateOnWrite	Yes	cluster	Yes	autoRepOnWrite
scsp.requireExplicitContextCreate	Yes	cluster	Yes	requireExplicitContextCreate
search.enableCustomMetadataTyping	Yes	cluster	Yes	enableCustomMetadataTyping
search.enableDelimiterPaths	Yes	cluster	Yes	enableDelimiterPaths
security.administrators	Yes	cluster	Yes	addModifyAdministrator, removeAdministrator
security.secureLogging	Yes	cluster	Yes	secureLogging
snmp.timeout	Yes	cluster	Yes	snmpTimeout

Configuring an Rsyslog Server

Configure an rsyslog server running the RHEL or CentOS operating system to accept incoming syslog messages from Swarm.

See the [rsyslog man page](#) or the [rsyslog documentation](#).

To configure the syslog server:

1. Log in as a user with root privileges.
2. Execute the following command:

```
vim /etc/rsyslog.conf
```

3. In the **rsyslog.conf** file, comment out the following lines to accept inbound UDP connections on port 514:

```
$ModLoad imudp.so
$UDPServerRun 514
```

4. Edit the file so the timestamp and IP address of incoming syslog messages appear.
5. Locate the following text:

```
#### GLOBAL DIRECTIVES ####
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat
```

6. Change this text to the following:

```
#### GLOBAL DIRECTIVES ####
$template myFormat,"%fromhost-ip% %rawmsg%\n"
$ActionFileDefaultTemplate myFormat
```

7. (Optional) Create a log file for each Swarm product by configuring a log file per logging facility:

```
local5.* /var/log/caringo/cr.log
local6.* /var/log/caringo/castor.log
```

8. (Optional) Create a log file based on any desired string in the log message using the `:msg` parameter. For example, to create a log file that only includes messages with the word "Trims", use this format:

```
:msg,contains,"Trims" /var/log/caringo/trims.log
```

The result matches the following messages:

```
2016-02-11T17:06:10.359Z 10.1.1.153 [21] debug : 00:51,602 HP
DEBUG: Trims decidable locally / trims needed: 0/0
2016-02-11T17:06:10.359Z 10.1.1.153 [21] debug : 00:52,484 HP
DEBUG: Trims decidable locally /trims needed: 0/0
```

9. Check **iptables** and Security-Enhanced Linux (SELinux) to verify inbound port 514 is not blocked.
10. Restart the rsyslog process:

```
service rsyslog restart
```

Configuring the Overlay Index

- [Requirements](#)
- [Determining status](#)

Swarm has an Overlay Index, which provides increased scalability in a storage cluster. When enabled, the Overlay Index tracks object locations in a *shared cluster-wide index*, minimizing multicast traffic in the cluster network.

The Overlay Index creates a dynamic master index in RAM from the local node indexes of all existing objects in the cluster. The Overlay Index locates the nodes containing the targeted object and directs the request to an appropriate node without multicasting to locate it when an SCSP request for an existing object is sent to a Swarm storage cluster. This process minimizes the multicast traffic and associated processing in a storage cluster.

The knowledge of existing nodes in the cluster is refreshed regularly and the Overlay Index evaluates whether changes to its model of the cluster are required every 5 minutes, verifying the Overlay Index is optimally spread across all available nodes in the cluster as new nodes are added or existing nodes are removed from the cluster. Adjustments to the Overlay Index's distribution due to changes in the nodes participating in the Overlay Index (particularly immediately after a cluster reboot) happen quickly but are not instantaneous. Distribution adjustments should not affect client activity.

In the Swarm UI, the Overlay settings appear under the **Index** section of the **Cluster Settings** page:

Index

index.optimize404 true false

index.ovMinNodes

index.overlayEnabled true false

By default, the Overlay Index is enabled for clusters with 3 or more nodes. the Overlay Index can be configured without rebooting the cluster by setting it as above in **Cluster Settings** or through its SNMP OIDs:

Overlay Settings	Default	SNMP OID	Description
index.optimize404	TRUE	overlayOptimize404	Optional. Enables the Optimize 404 feature in the overlay index, which returns 404 without multicast where possible. With the Overlay Index enabled and fully populated and Optimize 404 enabled, Swarm provides faster 404 (Not Found) responses for missing objects.
index.overlayEnabled	TRUE	overlayIndexEnabled	Enables the overlay index.
index.ovMinNodes	3	overlayMinNodes	Count. The minimum number of cluster nodes needed to activate use of the overlay index.

Requirements

The Overlay Index provides these features:

- Enables the Health Processor to quickly discover and remove over-replication generated by failed volume recovery (FVR) and erasure coding recovery (ECR), which maximizes the disk space on cluster nodes.

- Reduces the multicast traffic in a storage cluster network.
- Provides faster 404: Not Found response time for nonexistent immutable and aliased objects when Optimize 404 is enabled.

To support these features, the Overlay Index requires additional index slots in cluster node RAM to store the additional index information. The Overlay Index is not populated and the cluster continues to use multicast to locate objects if there are not enough nodes or RAM resources to hold the additional index information. Additional RAM *must* be added and the nodes rebooted to take advantage of the Overlay Index if the current node indexes are full.

See the [Hardware Requirements for Storage](#) for RAM requirements.

Determining status

Check the value of the indexOverlayStatus SNMP OID to determine the current state of the Overlay Index at a given time. The status of the Overlay Index is one of the following:

- **Disabled:** When the overlay is turned off.
- **Uninitialized:** When the overlay is first activated.
- **Operational:** When population of the Overlay Index is in process.
- **Authoritative:** When the node servicing the SNMP query has determined the Overlay Index is fully populated for the entire cluster. The **Optimize 404** feature only works when the Overlay Index is authoritative.

During normal operation, the state of the Overlay Index toggles between "authoritative" and "operational" as the structure of the cluster changes with new nodes being added or existing nodes being removed. The status may also indicate a variety of reasons why the Overlay Index is not operational, such as insufficient nodes or inadequate Overlay Index space.

Configuring Encryption at Rest

- [About Encryption at Rest](#)
- [Best Practices](#)
- [Encryption Settings](#)
- [Generating Encryption Keys](#)
- [Enabling Encryption on a Cluster](#)
- [Encrypting Existing Swarm Volumes](#)
- [Troubleshooting Encryption](#)
- [Disabling Encryption on a Cluster](#)
- [Decommissioning an Encrypted Cluster](#)

About Encryption at Rest

Swarm provides an option to encrypt all user data on drive volumes. Swarm encrypts the data as it writes it to the drive and decrypts it on access. Because this occurs down at the kernel level, the effect is invisible: there is no difference in accessing encrypted versus unencrypted objects. Encryption is controlled entirely through `[disk]` settings in the configuration, but these cannot be changed dynamically (using the Swarm UI or SNMP).

What it protects – Swarm volumes generally contain sensitive and proprietary client information. Implementing encryption at rest provides two types of protection:

- Security for the data on all removed and failed physical drives.
- The ability to render all data in a cluster inaccessible by purging the encryption key.

Important

Swarm data is only encrypted when “at rest” (stored physically on the Swarm drives). The following is not encrypted:

- objects within Swarm memory
- objects in network transit between Swarm volumes or clusters
- object metadata sent to Elasticsearch for indexing

What it does – Encryption exists at the drive level, not during transmission. This is how encryption at rest works:

- Data on each new drive is encrypted with an administrator-supplied key Swarm accesses from the `volume.cfg/cluster.cfg` file (Swarm does not persist any copies of the keys within the cluster).
- The data on an encrypted drive removed from a chassis cannot be accessed anywhere, without the administrator-supplied key.
- Swarm can access the data on an encrypted drive moved to another cluster by using the administrator-supplied key.
- All data on encrypted Swarm volumes are permanently and safely inaccessible if the administrator-supplied key is destroyed.
- There is no method for recovering data if the administrator-supplied key is lost.

Caution

DataCore cannot restore lost keys, and there is no master key. Swarm cluster administrators are solely responsible for managing these keys, and they must protect the key by storing a copy in a safe and secure location.

Key concepts— Volumes encrypting data make use of encryption keys, and Swarm does not reformat existing Swarm volumes to enable encryption. Here are some implications:

- **Existing volumes** — *Nothing changes until adding a new volume on a node with the new settings* if encryption is enabled on an existing storage cluster. For Swarm to reformat a volume for encryption, the node must be rebooted with the new settings and the volume must be a non-Swarm (new or retired) drive.
- **Mixed volumes** — It is no trouble having a blend of encrypted and unencrypted volumes in a cluster and within a given node.
- **Converting volumes** — To have encryption implemented across an existing cluster, reboot the nodes with the new settings and systematically retire, reformat, and add back the volumes, at which time Swarm sets up the hardware for encryption. See [Retiring Hardware](#).
- **Replication** — Because encryption is low-level (local to the drive), it has no effect on replication (or any other request), and the key is not needed by the target cluster. Each source volume, if encrypted, has the key needed to read its *own* objects and send them to the remote cluster. Whether replicated objects are encrypted is completely independent of how they are stored in the source.

Example — Suppose there are three nodes in an existing source cluster and it is decided to enable encryption at rest. Update the configuration with the needed settings. Add a fourth node (Node 4) to the cluster and boot it individually. Nodes 1 through 3 do not have the encryption settings because they have not been rebooted, but Node 4 does have the settings to apply to new volumes. Since this node has booted with fresh, non-Swarm drives, Node 4's volumes are *all* encrypted at rest. For each request Node 4 receives, it uses encryption keys to write and read from the encrypted volumes. That is, if Node 3 needs to write an object to Node 4, Node 4 uses keys to store the object with encryption automatically. Given a remote replication cluster, the principle is the same. The source cluster (with Node 4) tells the target cluster to fetch specific objects for replication. Node 1 at the target cluster asks Node 4 at the source cluster for some objects. Since Node 4 has the keys, it can read the object from a drive and return the object to Node 1 in the target cluster. How Node 1 in the target cluster stores the object is up to *that* node's configuration. Because encryption is completely local, the target cluster volumes are encrypted if set up.

Performance impact — Encryption while reading and writing is a CPU-intensive activity and can typically expect to see a 10-30% performance overhead depending upon workload and hardware. The 2010 Intel Core processor family and later include special AES-NI instructions that implement the more complex and performance intensive steps of AES encryption. These instructions are implemented by AMD in processors starting late in 2011. Swarm's kernel takes advantage of the AES-NI instruction set if available in the CPU.

For more information, see [Intel Advanced Encryption Standard Instructions](#) and [Wikipedia AES instruction set](#).

 **Tip**

To determine if a given processor has AES-NI support, run `grep aes /proc/cpuinfo` from a Linux command shell.

Best Practices

- Approach encryption as a cluster-wide setting. Although it is possible to enable encryption separately on each chassis (each has a separate `volume.cfg`), use the same encryption settings cluster-wide.
 - *Exception:* Use chassis-specific encryption keys, if needed.
- Provide physical security measures to verify encryption keys are protected (on the physical chassis with USB sticks, or on the CSN) from both theft and loss.
- Update the configuration settings and then systematically retire and reformat existing drives to verify an entire cluster is encrypted.
- Systematically retire and reformat existing drives so they use the new key to change encryption keys.
- [Elasticsearch recommends](#) implementing the encryption method **dm-crypt** to guarantee Elasticsearch metadata content is also encrypted. Swarm does not automate or manage the encryption key process for Elasticsearch volumes.

Encryption Settings

Volume encryption settings and keys are configured through **[disk]** settings in the volume.cfg/cluster.cfg file.

[disk] Setting	Default	Description
encryptNewVolumes	0 (False)	Required. Boolean. The feature must be explicitly enabled. Indicates new Swarm volumes should be encrypted with the specified primary key. Set <code>disk.encryptedKeyPrimary</code> if set to 1 (True).
encryptionKeys	{ }	Required. A dictionary of key name/value associations. Swarm uses them to read volumes encrypted with these keys. Example: <pre>{ 'cluster_key_5_15_2016': 'a24f8ec391ab3341', 'cluster_key_5_12_2015': 'de3498245ce8bf89' }</pre> Use the <code>disk.encryptedKeyPrimary</code> setting to select which key in <code>disk.encryptedKeys</code> is "primary". The other keys are considered to be "secondary", and they are only used to read/write volumes already encrypted.
encryptionKeyPrimary	empty string	String. Contains the name of the encryption key to be used to format new encrypted Swarm volumes. Required: <ul style="list-style-type: none"> <code>disk.encryptNewVolumes</code> <i>must</i> be True. <code>disk.encryptNewVolumes</code> <i>must</i> be False if <code>encryptionKeyPrimary</code> is blank. <code>disk.encryptedKeyPrimary</code> <i>must</i> match one of the key names in <code>disk.encryptedKeys</code> if <code>disk.encryptedKeyPrimary</code> is set, so copy the key name exactly.
encryptionCipher	aes-xts-plain64	String. Indicates the preferred cipher algorithm to use when formatting new volumes. Keep the default unless directed otherwise. Valid values: aes-xts-plain64, aes-cbc-essiv
encryptionKeySize	512	Integer (in bits). Indicates the preferred key size to use when formatting new volumes. Keep the default unless directed otherwise. Valid values: 128, 256, 512
encryptionHash	sha512	Indicates the preferred hash algorithm to use when formatting new volumes. Keep the default unless directed otherwise. Valid values: sha256, sha512
encryptionIterationTime	5000	Integer (in milliseconds). Indicates the maximum iteration time LUKS uses to derive a key to use when formatting new volumes. Keep the default unless directed otherwise. Valid values: positive integers ≥ 1000

Generating Encryption Keys

Create encryption keys using tools available from the standard Linux and Windows OS package repositories.

Here is an example script for generating a key:

Generate encryption key

```
#!/usr/bin/python
import random
random.seed()
print "%064x" % random.getrandbits(512)
```

Note: 64 hex digits is 256 bits, and 128 hex digits is 512 bits.

Enabling Encryption on a Cluster

Swarm begins encrypting every *new* (not formatted by Swarm) volume it detects and formats when enabling encryption for a cluster and setting up the required encryption keys.

To implement encryption across a cluster, do the following:

1. Open the cluster configuration (`node.cfg/cluster.cfg` file) for editing.
2. Add appropriate values for these settings:

```
[disk]
encryptNewVolumes = true
encryptionKeys = {'key_2018-03-19': 'a24f8ec391ab3341', 'key_2016-09-27':
'de3498245ce8bf89'}
encryptionKeyPrimary = key_2018-03-19
```

3. **Important:** *Secure copies of the encryption keys.*
4. Reboot the cluster to activate the settings change.
5. Add any new hardware, which Swarm formats for encryption.



Important

Any existing unencrypted Swarm volumes remain unencrypted, regardless of any hot-plugging performed with them within the cluster. They remain unencrypted and accessible without encryption keys. See next.

Encrypting Existing Swarm Volumes

Reformat and remount the volumes if retiring volumes to implement encryption at rest. Contact DataCore Support for a utility to streamline this process. (v10.1)

Alternatively, see [Encrypting Existing Swarm Volumes Manually](#).

Troubleshooting Encryption

Unmountable Volumes

This is how Swarm handles volumes it cannot mount:

Encrypted Swarm volume Swarm cannot open	This occurs if the key is missing. Swarm ignores the volume and logs an error.
Encrypted non-Swarm volume	Swarm ignores the volume and does <i>not</i> format it.
Unencrypted non-Swarm volume	<p>A 60-second countdown timer is displayed on the physical console to provide time to prevent erroneous formatting if non-blank storage volume that appears to have data from other operating systems is detected when a Swarm Storage node boots. The most likely scenario for this happening is if a server or laptop is inadvertently PXE booted from the storage network. The countdown timer provides the opportunity to power off the system before the drive is formatted.</p> <p>After the 60-second countdown timer expires, Swarm formats the volume.</p>

Encryption Status and Logging

To see the encryption status of the volumes, view the SNMP volumes table for the cluster. Alternatively, view the status on the [Chassis Details](#) page of the Storage UI or the [Node Status](#) page of the legacy Admin Console: Swarm puts "(encrypted)" after the volume ID of volumes that are encrypted. Swarm writes a console message if it cannot open a volume because of a problem with the encryption key.

On startup or hotplug events, this is how Swarm logs encryption:

- Each non-encrypted drive that is mounted is logged to the console, "Mounted non-encrypted volume /dev/sda" if encryption is enabled (`encryption.primaryKey` is set). An error is logged to syslog.
- An error is logged to the console, "Unable to mount encrypted volume /dev/sda" if an encrypted volume cannot be mounted (such as for a missing key). An error is logged to syslog.
- The log entry of that volume includes the encryption status of the volume when a volume is mounted.
- During hotplug events, a volume with a non-Swarm encrypted partition is mounted and formatted as a Swarm volume immediately.

Disabling Encryption on a Cluster

The change affects how Swarm formats any new volumes it detects when disabling encryption on a cluster. Existing encrypted volumes, even if moved (hot plugged), remain encrypted and accessible only with encryption keys.

To remove encryption entirely from a cluster, do the following:

1. Edit the cluster configuration (.cfg) file.
 - a. Disable new volume encryption.
 - b. Remove the primary encryption key designation. (This makes it a *secondary* encryption key.)
2. Reboot all nodes requiring unencrypted volumes (to activate the settings change).
3. Systematically retire all encrypted volumes. Swarm relocates the data to other volumes.
4. Add back each volume.
 - a. Swarm formats each new volume as unencrypted and mounts it when detected.
 - b. As new data fills up each volume, it is unencrypted and requires no key.

Decommissioning an Encrypted Cluster

Perform the following to decommission an encrypted cluster, guaranteeing none of the encrypted data is ever retrievable:

1. Delete the encryption keys from the cluster configuration (.cfg) file.

2. Destroy all copies of the encryption keys.

Reminder: `disk.encryptionKeys` can have more than one key value, and *any* key values can be used to open an encrypted Swarm volume.

3. Reboot the cluster.

Without the keys, Swarm cannot mount the volumes, so all are out of service.

4. Remove, reformat, and repurpose the volumes.

Configuring Cluster Policies

- [Replication Policy](#)
- [Erasure Coding Policy](#)
- [Versioning Policy](#)

There are three types of storage policies in Swarm: *replication*, *erasure coding*, and *versioning*. They can be customized at the level of domains and buckets, but this section concerns the Swarm settings that control the cluster-wide requirements. They appear in the **Policy** section of the **Cluster Settings** in the Swarm UI:

Policy		
policy.eCEncoding	5:2 anchored	Default: unspecified anchored
policy.eCMinStreamSize	1Mb	Default: 1Mb anchored
policy.replicas	min:2 max:16 default:2 anchored	
policy.versioning	allowed	Default: disallowed

Those settings that show an SNMP name are [persisted settings](#). They can be update dynamically without a cluster restart.

See [Swarm Storage Policies](#).

Replication Policy

See [Implementing Replication Policy](#) for how to create custom replication policies on specific domains and buckets.

Setting	Default	
policy.replicas	min:2 max:16 default:2	The minimum, maximum, and default number of replicas allowed for objects in this cluster. Can differ from the policy in a replicated target cluster. Examples: min:2 max:16 default:3 min:3 max:10 default:3
SNMP: policyReplicas	anchored	

Erasure Coding Policy

See [Implementing EC Encoding Policy](#) for how to create custom EC encoding policies on specific domains and buckets.

Setting	Default	
ec.conversionPercentage	0	Percentage, 1-100; 0 stops all conversion. Adjusts the rate at which the Health Processor consolidates multi-set erasure-coded objects each HP cycle. Lower to reduce cluster load; increase to convert a large number of eligible objects faster, at the cost of load on the cluster. If enabled, requires policy.eCEncoding to be specified.
SNMP: ecConversionPercentage		

<p>ec.maxManifests</p>	<p>6</p>	<p>Range, 3-36. The maximum number of manifests written for an EC object. Usually, p+1 are written for a k:p encoding.</p> <p><i>Requirement:</i> Manifests must all be written to different nodes, even when using ec.protectionLevel=volume.</p> <p>Do not set above 6 unless directed by Support.</p>
<p>ec.minParity</p>	<p>-1</p>	<p>Range -1 or 1-4; default of -1 is max(policyminreps - 1, 1), where policyminreps is the min value in policy.replicas. The minimum number of parity segments the cluster requires. This is the lower limit on p for EC content protection, regardless of the parity value expressed on individual objects through query arguments or lifepoints.</p>
<p>ec.protectionLevel</p> <p>SNMP: ecProtectionLevel</p>	<p>node</p>	<p>Either 'node', 'subcluster', or 'volume'. At what level segments must be distributed for an EC write to succeed. Note: multiple segments are allowed per level, if needed. 'node' (default) distributes segments across the cluster's physical/virtual machines. 'subcluster' requires node.subcluster to be defined across sets of nodes. You must have (k+p)/p nodes/subclusters for those levels; at minimum, you must have k+p volumes.</p> <p>See details below.</p>
<p>ec.segmentConsolidationFrequency</p> <p>SNMP: ecSegmentConsolidationFrequency</p>	<p>10</p>	<p>Percentage, 1-100, 0 to disable. How quickly the health processor consolidates object segments after ingest. Increase this value (such as to 25, to consolidate over 4 HP cycles) to make new content readable sooner by clients. For multipart uploads via S3 clients, 10 is recommended; for SwarmNFS, 100 is recommended, with extra space allowances for trapped space.</p> <p>Consolidation changes the ETag (which affects If-Match requests) and Castor-System-Version headers, but Content-MD5 and Composite-Content-MD5 headers are unchanged. Therefore, have clients use the hash and last-modified date, rather than ETag, to find if an object has changed.</p>
<p>ec.segmentSize</p>	<p>-1</p>	<p>In bytes; default of -1 implies 200 MB, with recommended minimum of 100 MB. The maximum size allowed for an EC segment before triggering another level of erasure coding. For mostly large (1+ GB) objects, increase to minimize the number of EC sets, which reduces index memory usage. Alternatively, increase the size as needed per write request using the 'segmentsize' query argument.</p>
<p>policy.eCEncoding</p> <p>SNMP: policyECEncoding</p>	<p>unspecified anchored</p>	<p>The cluster-wide setting for the EC (erasure coding) encoding policy. Valid values: unspecified, disabled, k:p (a tuple such as 5:2 that specifies the data (k) and parity (p) encoding to use). Add 'anchored' to set this cluster-wide; remove it to allow domains and buckets to have custom encodings.</p> <p>Examples:</p> <p>5:2</p> <p>6:3 anchored</p>

policy.ecMinStreamSize SNMP: policyECMinStreamSize	1MB anchored	In integer units of megabytes (MB) or gigabytes (GB); must be 1MB or greater. The size that triggers an object to be erasure-coded, if specified (by eCEncoding, lifepoint, query argument) and allowed by policy. Below this threshold, objects are replicated unless they are multipart or chunked writes. Add 'anchored' to set this cluster-wide; remove it to allow domains and buckets to have custom values. Examples: 100Mb 1GB anchored
---	--------------	---

What EC Protection Level is needed?

The EC protection level determines how strictly EC segments must be distributed for a write to succeed, or else return an error (412 Precondition Failed) to the writing application. After Swarm writes an object to the cluster, the health processor attempts to maintain the requested protection level. If cluster resources become unavailable, it degrades gracefully. When this occurs, the health processor logs errors, alerting you that the requested protection cannot be maintained and data may be at risk.

Regardless of the protection level set, Swarm always makes a best effort to distribute segments as broadly as possible across hardware, to protect data.

ec.protectionLevel	Cluster requirements	Effect
subcluster	$\geq (k+p)/p$ subclusters	Requires a subcluster for every p segments. Use only if geographical or systems-based subclusters are defined to factor into content protection.
node (default)	$\geq (k+p)/p$ nodes	Requires a node for every p segments. Use for most situations. Important: When working with a small number of nodes, verify the EC encoding can support what exists. <ul style="list-style-type: none"> With 3 nodes, use 3 : 2 encoding ($(3 + 2) \div 2 = 3$ nodes required), but not 3 : 1 encoding ($(3 + 1) \div 1 = 4$ nodes required). With 4 nodes, use 4 : 2 encoding ($(4 + 2) \div 2 = 3$ nodes required), but not 4 : 1 encoding ($(4 + 1) \div 1 = 5$ nodes required).
volume	$\geq k+p$ volumes	<i>Least protection.</i> Requires $k+p$ volumes, but $p+1$ nodes are still needed because the manifest must be written to separate nodes. Use only if you have insufficient nodes for node-based protection.
	$< k+p$ volumes	<i>Unsupported.</i> EC writes fail.



Deprecated

The setting `ec.subclusterLossTolerance` has been deprecated and needs to be removed from configurations when upgrading to Swarm 10.

Versioning Policy

Swarm has policy support for object versioning. Versioning can be enabled for specific contexts (domains and buckets) after the cluster is configured to permit versioning of objects.

See [Implementing Versioning](#) for how to create versioning policies on specific domains and buckets.

Setting	Default	
<p>policy.versioning</p> <p>SNMP: policyVersioning</p>	<p>disallowed</p>	<p>Specifies whether versioning is allowed to be enabled on contexts (domains and buckets) within the cluster. Valid states: disallowed, suspended, allowed. This policy overrides context-level policies. Disallowed removes historical versions, if any. Suspended stops creation of new versions but retains version history.</p> <p>Examples:</p> <p>allowed</p> <p>disallowed</p> <p>suspended</p>

Configuring Volumes Options

- [disk.volumes setting](#)
- [device option](#)
- [policy option](#)

The `disk.volumes` option in the node or cluster configuration file specifies the volumes used by Swarm. This specification includes the device names and optional flags for handling these volumes. One `disk.volumes` definition is allowed in a configuration file. This setting *cannot* be changed dynamically (in the Swarm UI or SNMP).



Warning

Swarm erases any non-Swarm data on all volumes it uses. For best results, run Swarm only on nodes that are free of non-Swarm data.

disk.volumes setting

Best practice – Use `disk.volumes = all`, which enables Swarm to use all volumes greater than the configured `disk.minGB` (64 GB by default). Configure Swarm to format and mount smaller disks by decreasing the value of `disk.minGB`. A USB flash drive is automatically excluded from the volume list if booting Swarm from one.

Additionally, `disk.volumes = all:`

- Supports hot plugging
- Supports exceptions (`all except`)
- Supports `:k` (keep policy)

volumes syntax

```
disk.volumes = volume-specification
volume-specification ::= all-volumes | volume-list | ''
all-volumes ::= 'all' [ ':' policy ] [ space 'except' space device-list ]
device-list ::= device [ space device [ ... ] ]
device ::= Linux-device-path
space ::= space or Tab character and not the word space
policy ::= 'k'
```

Example disk.volumes entries

```
disk.volumes = all
disk.volumes = all:k
disk.volumes = all except /dev/hda
disk.volumes = /dev/sda /dev/sdb
disk.volumes = /dev/hda:k /dev/hdc:k
disk.volumes =
```

No volumes – If `disk.volumes` is set to an empty string, a diskless machine is specified, so Swarm does not mount any volumes. If the `disk.volumes` setting itself is absent, Swarm mounts all available volumes (equivalent to `disk.volumes=all`).

device option

The device component is either the keyword `all` or the Linux device path string for the disk. When using the keyword `all`, do not include any other device path specifications unless they follow `except`.

Best practice – Use `disk.volumes = all` and avoid Linux device paths. Exclude specific volumes from being formatted and used by Swarm by listing them after `except`:

```
disk.volumes = all except /dev/sda dev/sdb
```



Important

Only use `except` with `disk.volumes = all`.

Example Linux device paths

- `/dev/hda`
- `/dev/sda`
- `/dev/sdb`

Older IDE disks, also known as [Parallel ATA](#), [EIDE](#), ATA-33, ATA-66, ATA-100, or ATA-133, use **hd** device names. These disks are configured as master or slave devices on each IDE controller. Typically, the master devices are `/dev/hda` and `/dev/hdc`, while the slave devices are `/dev/hdb` and `/dev/hdd`.

[SCSI](#), [SAS](#), and [SATA](#) disks typically use **sd** device names. The device letters are assigned sequentially in the order in which the disks are discovered starting at `/dev/sda`. The hardware report in the utility menu shows the actual names in use on a node.

If an invalid device name is specified in the device component, the node log indicates an error during the format operation. Incorrectly-specified volumes are not used.

policy option

The policy option allows instructing Swarm how to handle a volume. Currently, these handling features involve the formatting characteristics of the physical device.

The format policy allows overriding the default volume expiration behavior by specifying the **:k** (keep) policy.

See [Returning a Stale Volume to Service](#).

Configuring an External Time Server

- [Guidelines for Time Servers](#)
- [Configuring a Node with NTP](#)
 - [Important](#)
- [Configuring a Node without NTP](#)
 - [Warning](#)

Precisely synchronized time is critical to the integral processes in the Swarm storage cluster, such as versioning, lifepoints, and updates. Unexpected results may occur if the nodes in a storage cluster are not synchronized with each other.

Specify NTP ([Network Time Protocol](#)) servers via the `network.timeSource` setting, which cannot be changed dynamically (using the Swarm UI or SNMP).

Guidelines for Time Servers

Swarm requires extremely precise clock synchronization to prevent data loss. Follow these guidelines to guarantee adequate synchronization:

- **Use NTP servers.**
Best practice is to use NTP servers to synchronize the clock in each cluster node.
- **Do not use OpenNTPD or SNTP.**
Swarm supports the NTP protocol. The [Open Network Time Protocol Daemon](#) (OpenNTPD) and the [Simple Network Time Protocol](#) (SNTP) are not supported because these protocols do not implement high-accuracy timing methods required by Swarm.
- **Use trusted NTP servers.**
Use trusted NTP servers, whether they are dedicated hardware solutions in the internal network or external, public servers.
- **Synchronize client systems.**
Swarm does not synchronize the client system clocks. Best practice is to synchronize these clocks with the NTP servers as well.

Configuring a Node with NTP

To configure a node to use NTP, populate the `network.timeSource` setting in the node or cluster configuration file, using one or more IP addresses or host names based on the DNS server configuration in the network.

 **Important**

Verify the `network.timeSource` setting is correct *and* that the nodes have network access to the NTP pool servers. Nodes time out waiting for a connection and automatically restart if they cannot reach public NTP servers.

If node is *not* configured to a DNS server

Set the `network.dnsServers` setting to a valid value, and set the `network.timeSource` setting to one or more IP addresses. See the [Settings Reference](#).

```
network.dnsServers = 84.200.69.80 84.200.70.40
network.timeSource = 192.168.0.20 192.168.0.50 192.168.0.110
```

<p>If node uses DHCP that provides a DNS server</p>	<p>Set the <code>network.timeSource</code> setting to one or more host names. Decide to use public NTP pool servers or internal NTP servers.</p> <p>The NTP Project recommends using pool servers that are close to the servers' time zone. See the page, How do I use pool.ntp.org?</p>
<p>If node uses U.S.-based pool servers</p>	<p>Use the following setting value:</p> <pre>network.timeSource = 0.us.pool.ntp.org 1.us.pool.ntp.org 2.us.pool.ntp.org</pre>

Configuring a Node without NTP

Do not run a storage cluster in a production environment without using NTP. In demonstration or development environments where there is no internal or external NTP server available, choose a minimum of one or a maximum of two nodes as the master clock and synchronize the clocks in the remaining nodes to the master clock in one of those nodes.



Warning

Verify the BIOS clocks in all new nodes are set relatively close to the correct [GMT](#) time before they join the cluster if creating a Swarm storage cluster without an external NTP time source.

All Swarm node clocks are set to GMT (not local time), and they do *not* change for [daylight saving time](#).

To implement a cluster without using NTP:

- Set the following setting in the configuration file of any single node in the cluster:

```
network.timeSource = system
```

All other nodes in the cluster attempt to synchronize clocks to this node.

Configuring External Logging

- [Obscuring UUIDs in Logs](#)
- [Configuring the Logging Host](#)

This section deals with setting up the `[log]` section of the configuration.

Obscuring UUIDs in Logs

The `log.obscureUUIDs` parameter allows control whether entire UUIDs display in logs.

With `log.obscureUUIDs = false` (the default), the entire UUID displays as follows:

```
<183>2016-02-11T17:06:10.359Z SCSP DEBUG: REQUEST: GET 172.16.0.33 <None>/358a76f06ffe7e4128d5e6c  
  alias=true (request:3087553833379006269 connection:25579392) 172.16.0.32 14/11 20:08:37.065
```

With `log.obscureUUIDs = true`, 20 of the 32 characters of the UUID display, as follows:

```
<183>2016-02-11T17:06:10.359Z SCSP DEBUG: REQUEST: GET 172.16.0.33 <None>/358a76f06f...c467a2a96d  
  alias=true (request:3087553833379006269 connection:25579392) 172.16.0.32 14/11 20:08:37.065
```

Keeping the default values makes issues easier to troubleshoot. Set `log.obscureUUIDs = true` if there are concerns about the security implications of displaying entire UUIDs in the logs.

Configuring the Logging Host

The **log.host** option enables Swarm log messages to be sent to a central syslog, rsyslog, or syslog-ng server. The examples here show the specific Swarm items that need to be added to the UNIX configuration files for syslog or syslog-ng. The actual configuration files likely contain additional information for logging messages from other hosts and other logging facilities.

See the [syslogd reference](#) and the [syslog-ng reference](#).

To configure the Syslog server:

1. Set the **syslog** or **syslog-ng** daemon options to enable logging from a remote host.

Edit `/etc/sysconfig/syslog` to the **SYSLOGD_OPTIONS** to the following:

```
SYSLOGD_OPTIONS="-r -m 0"
syslog-ng
```

2. Configure other logging options.

This example shows a sample configuration with the standard syslog program. When editing the `syslog.conf` file, **facility.level** must be followed by a tab character to separate it from the destination specification. (See the [syslog.conf manual page](#).)

```
# /etc/syslog.conf
local6.* /var/log/castor.log
syslog-ng
```

This example shows a sample syslog-ng configuration. (See the [syslog-ng.conf manual page](#).)

3. Add the following in the **filters** section:

```
#CAstor filter
filter f_castor {facility(local6)};
```

If a **filters** section does not exist, add one after the **source** statement and before the **destinations** section.

4. Add the following in the **destinations** section:

```
#CAstor destination
destination d_castor {file("/var/log/castor.log")};
```

5. Add the following **log** statement:

```
#Caringo-specific additions #
log {source(s_net){ udp()}; filter(f_castor); destination(d_castor)};
```

The statement `(source(s_net){ udp()};` is the default remote service source for syslog-ng. If using a different remote source name, replace `(source(s_net){ udp()};` with the name used.

6. Enter the following commands to restart the service with the changes:

```
sudo /etc/init.d/syslog-ng stop
sudo /etc/init.d/syslog-ng start
```

7. Set Swarm logging options in the node or cluster configuration file.

- **log.host.** The fully qualified host name or IP address of the syslog or syslog-ng server.
- **log.port.** The server port.
- **log.level.** Set the log level below:

Level	Meaning
0	No logging

50	Critical messages and announcements only
40	Errors, critical messages, and announcements
30	Warnings, errors, critical messages, and announcements
20	Info, warnings, errors, critical messages, and announcements
10	Debug, info, warnings, errors, critical messages, and announcements (the most verbose log level)

Fully qualified parameters:

```
log.host = 192.168.0.100  
log.port = 514  
log.level = 40
```

Section/unqualified parameters:

```
[log]  
host = 192.168.0.100  
port = 514  
level = 40
```

Time of Last Access - atime

- [Implementing atime](#)
- [Configuring atime](#)
- [Using atime with SCSP](#)
- [Using atime with Elasticsearch](#)
- [Using atime with Content UI](#)
- [Limitations and Troubleshooting](#)

Swarm can capture and persist the *time of last access* ("atime") on objects and add it to the search feed. This allows search queries to list objects that may be candidates for deletion or tiering (moving to cheaper storage). Write an application using atime values to purge "cold" objects not read in the last three years. Swarm stores the atime as the `Castor-System-Accessed` header and indexes it as the `accessed` field in Elasticsearch, which is useful for bulk evaluations of content.



Performance impacts

Tracking atime does affect performance, so enable only if needed. Tracking access times can incur long-tail latencies on first reads, particularly when disk demands are heavy. For around 90% of objects read for the first time, the latency is negligible (<1 ms); when requests queue on specific volumes do the effects become noticeable. Subsequent reads *within* the window of the `disk.atimeGranularity` value have no performance impact. (SWAR-7772)

Having high numbers of small object reads (such as thumbnail images) can cause memory indexes to run full.

Implementing atime

Because support for "atime" involves changes to the underlying Elasticsearch schema, existing feeds cannot be restart after a Swarm upgrade, as the **written** and **accessed** fields are not populated for some records have the incorrect type.



Tip

The atime feature requires a rebuilding of the search index, so take the opportunity to [migrate to Elasticsearch 6](#) with the same reindexing.

1. For intensive READ access scenarios, provision additional memory to support the load on the in-memory index.
2. Finish installing the storage cluster to Swarm 10, and install the latest versions of the Swarm metrics and search RPMs in the Elasticsearch cluster.
3. Enable the cluster setting for the feature, which is disabled by default: `disk.atimeEnabled = true`
4. Create a new search feed, which uses the new Elasticsearch schema that supports atime.
5. Complete these steps to transition to the new search food if a previous feed exists :
 - a. After the new feed completes processing, make the new feed the **Primary**.
 - b. Pause the old feed.
 - c. Delete the old feed and the old index data after verifying the new feed is working as expected.

Configuring atime

The public settings for "atime" are dynamic. These values can be updated on one node and Swarm updates all others, and the values persist across reboots. Following are all settings that control the gathering of atime information:

Settings for atime	Default	Type	Description
<code>disk.atimeEnabled</code> SNMP: <code>accessedTimeEnabled</code>	False	bool	Whether to track the time of last access on GET requests, stored in the <code>Castor-System-Accessed</code> header and indexed as the search field 'accessed'. Increases the load proportionally to the load of GETs in the cluster.
<code>disk.atimeGranularity</code> SNMP: <code>accessedTimeGranularity</code>	86400	int	In seconds; defaults to 1 day. The window of time during which atime is not updated. Multiple reads may have occurred within window of time. Lowering the value affects GET performance. A 1-second granularity provides most accurate accessed time results, but results in a GET performance penalty due to increased disk access.
<code>disk.atimeEnabledTime</code> SNMP: <code>accessedTimeEnabledTime</code>	0	float	<i>Non-UI.</i> Read-only. The Linux epoch timestamp recorded when <code>disk.atimeEnabled</code> was set to True. This time is nulled out in SNMP, REST API, and phone home reports if the atime feature is later disabled.

Using atime with SCSP

Swarm keeps a record of the request time of each object's last write or read (successful GET request) when enabling atime tracking for the cluster, and it sends that time to Elasticsearch as the **accessed** date field, for use in search queries. HEAD operations do not change an object's atime. To access atime without Elasticsearch, check the SCSP headers Swarm adds to the objects.

With atime enabled, both SCSP [HEAD](#) and [GET](#) requests include a **Castor-System-Accessed** header on the response when the **verbose** query argument is used. The **Castor-System-Accessed** response header has either the value of **Castor-System-Created** (because the object has not been read since the feature was enabled or the object was written) or else the read atime in the same GMT-based time format as `Castor-System-Created`. The 1-day granularity (default) in updating atime means additional reads may have occurred within that window of time.

Exceptions – GET requests trigger atime updates, *except* for these situations:

- Administrative and authorized admin requests
- Swarm requests for replication and other internal GET requests, such as for domains, settings, or manifests
- Any request with the special query argument to suppress recording atime: `notaccessed`
- Any request performing an integrity check or other specialized operation



Tip

The atime information is most useful on a HEAD request since the atime is returned without changing it. Although atime is returned on a GET request, it is simultaneously updated by the operation.

To determine if an object has been read, HEAD the object using the `verbose` query argument.

The **Castor-System-Access** value matches the **Castor-System-Created** if a read atime has not occurred:

```
> curl -I http://192.168.1.12:80/5647f528ea85667a44dc754f975816c6?verbose
HTTP/1.1 200 OK
Castor-System-Alias: 5647f528ea85667a44dc754f975816c6
Castor-System-Cluster: Baker
Castor-System-Created: Wed, 19 Jul 2017 17:42:48 GMT
Castor-System-Accessed: Wed, 19 Jul 2017 17:42:48 GMT
...
```

The **Castor-System-Access** value is more recent than the **Castor-System-Created** if a read has occurred:

```
> curl -I http://192.168.1.12:80/5647f528ea85667a44dc754f975816c6?verbose
HTTP/1.1 200 OK
Castor-System-Alias: 5647f528ea85667a44dc754f975816c6
Castor-System-Cluster: Baker
Castor-System-Created: Wed, 19 Jul 2017 17:42:48 GMT
Castor-System-Accessed: Tue, 02 Oct 2018 23:03:56 GMT
...
```

Using atime with Elasticsearch

In Elasticsearch, the atime value is indexed as the **accessed** date field, which can be used in Swarm Search [listing queries](#). Both the **written** and **accessed** fields are populated in the Elasticsearch record:

Metadata Field	Type	Description
accessed	date (written and listed as ISO 8601)	The date of last access appears in listing results if requested. The value does not reflect lifepoint conversion or segment consolidation that may have occurred. Matches the value for written until the first GET operation occurs, after which it updates for each qualified GET.
written	date (written and listed as ISO 8601)	Does not change for a particular object version (ETag).

Admin GET requests do not bump the atime value. Make SCSP [GET](#) requests with the **notaccessed** query argument, to suppress the atime update. This argument allows listing objects for management purposes without erroneously bumping the accessed date, as if an end-user or program had requested the object.

Argument	Value	Description
notaccessed	"yes"/"true"	Allows a GET request to complete without updating the accessed time on the object if the atime feature is enabled.

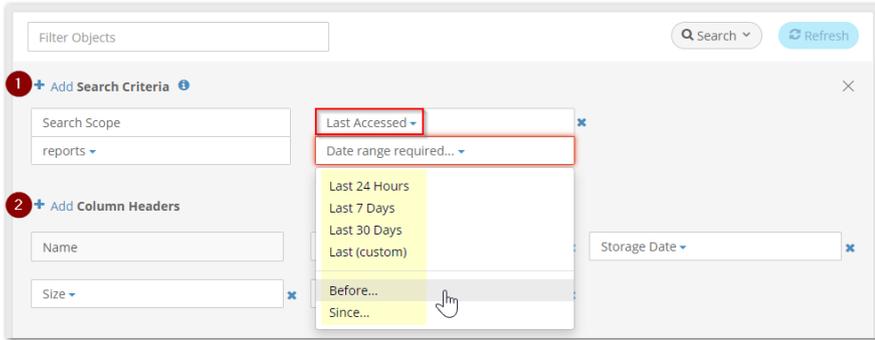
Using atime with Content UI

Show and filter on **Last Accessed** when adding columns and search criteria to the object filters, which is indexed in Elasticsearch as 'accessed'. Filtering on the time of last access includes standard and custom time spans from the present as well as **Before** and **Since** ranges. (v11.0)

Click the **Search** button to add filtering criteria once in a domain or bucket After **+ Add Search Criteria** for Last Accessed, **+ Add Column Header** for Last Accessed as well if the access date needs to appear in the results:

See [Search Collections](#).

Limitations and Troubleshooting



- The reported value may be stale if the feature was enabled and disabled repeatedly. To determine staleness, compare any value against the time when the atime feature was enabled: `atimeEnabledTime`.
- The reported atime, if present, is an atime defaulting to the granularity of a day; therefore, it is not necessarily the precise time (hours and minutes) of last access.

- Replication feeds do not get re-processed for atime changes.
- Swarm can lose a recent read atime for an object on that volume if a volume is lost.
- Stale atimes remain in the Elasticsearch records if the atime feature is disabled after having been enabled. Applications using this field need to check the state of the feature (whether `disk.atimeEnabled = True`) and when the feature was last enabled (`disk.atimeEnabledTime`).
- For monitoring, Swarm provides a per-node count of how many objects are assessed but whose atime is not propagated to Elasticsearch.

Configuring Power Management

Swarm includes an adaptive power conservation feature that supplements Swarm's naturally green characteristics. This power-saving mode, also referred to as Darkive™, spins down disks and reduces CPU utilization after a configurable period of inactivity.

In the Swarm UI, the options appear under the **Power** section of the **Cluster Settings** page:

Power

power.savingMode true false

power.sleepAfter

power.wakeAfter

Note

The configuration options for power management apply to the entire cluster.

How Power Management Works

The power-saving mode causes a node with no incoming SCSP requests (from clients or other Swarm nodes) in the last configurable **power.sleepAfter** seconds to change to an Idle status and pause its health processor. There is usually a delay between the *node* Idle status and its *volumes* Idle status because in-process replications are performed between nodes even after they are idle, to guarantee full data protection. After all queued activity is completed, the disks eventually spin down (if supported by the disk manufacturer) and display as Idle if no further activity causes disk I/O.

When it appears in the Swarm UI or Console, **Idle** has different meanings for a node and volume:

- **Idle node.** A node with no SCSP activity for a specified length of time. In an idle state, a node's health processor pauses, so an idle node is more likely to have idle volumes.
- **Idle volume.** A volume with no I/O activity for at least six minutes.

The cluster automatically wakes one or more nodes to carry out requests when needed and eventually revives all nodes if required. If there is no intervening activity after the configured **power.wakeAfter** period, the cluster wakes all nodes to perform data, disk, and replication integrity checks.

Power Management Settings

Adjust these settings to best suit your Swarm implementation:

<p>power. savingMode</p>	<p>Enables Power Saving Mode, which allows the system to sleep or power cap. Set to <code>False</code> to disable, which is called Full Performance Mode.</p> <p>A cluster with long periods of inactivity on nights and weekends can achieve significant power savings using this feature. Because only inactive nodes are affected, maximum available throughput is not affected, although additional latency is incurred on the first access to a node.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Best practice</p> <p>Use Full Performance Mode (<code>power.savingMode = False</code>) for these situations:</p> <ul style="list-style-type: none"> • The cluster is in constant use (24x7). • Uninterrupted feed restarts are critical for your operations. • The cluster is used for direct replication (DR). </div>
<p>power. sleepAfter</p>	<p>In seconds, 60 or greater; defaults to 2 hours. In Power Saving mode, how long a node is inactive before it becomes idle.</p> <p>This option determines how long a period of inactivity should occur with no incoming SCSP requests before Swarm pauses the node's health processor and displays it as Idle in the Swarm UI or Console. If you select Full Performance Mode, a node does not display as Idle. Setting the value to a small number allows nodes to become idle after a reasonable period of inactivity (two hours by default).</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Important</p> <p>Avoid setting this value to a long period, which <i>prevents</i> Swarm nodes from becoming idle and taking advantage of power savings.</p> </div>
<p>power. wakeAfter</p>	<p>In seconds; defaults to 8 hours. In Power Saving mode, how long a node is idle before it becomes active again.</p> <p>This option determines how long a node remains idle before Swarm wakes up an idle node to allow the health processor to validate disk content integrity and replicas. Setting the number to a small number reduces power savings, although the volumes and nodes return to an Idle state if additional client activity is not received.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Important</p> <p>Avoid setting this value too close to the power.sleepAfter value, which makes the disks cycle quickly between sleeping and waking and reduce power savings.</p> <p>Avoid setting this value to a long period, which can put content at risk because the health processor cannot run on sleeping nodes. See health.startDelay for more about the health processor.</p> </div>

**node.
archiveMode**

Disabled by default, which is the normal operating state. Set to TRUE to put the node into archive mode, where it remains idle in low-power mode without participating in cluster activity until its capacity is needed. Use this to provision new nodes into the cluster proactively, well before they are needed.

Archive Mode allows designating a new or empty node as an archive node so it remains idle in low-power mode without participating in cluster activity until its capacity is needed. This allows keeping additional storage capacity online and available without paying for the associated power costs of the additional nodes. Once an archive node is activated by diminishing capacity in the remaining cluster, it attempts to aggressively fill itself to capacity with incoming write requests. Once its storage capacity is full, the node returns to an idle state until its stored content needs to be read.

Configuring Content Integrity Settings

Use the following set of [configuration settings](#) to enforce your content integrity policy.

scsp.replicateOnWrite

This option allows an administrator to force a second replica of an object to be written to a different node prior to returning a success status to the client. If either write operation fails, the client receives an error status.

In the Swarm UI, it appears under the **SCSP** section of the **Cluster Settings** page:



scsp.replicateOnWrite true false

See [Configuring Replicate On Write](#).

scsp.validateOnRead

This option allows an administrator to require validation of all content reads before returning a successful read completion.

In the Swarm UI, it appears under the **SCSP** section of the **Cluster Settings** page:



Although this option can be specified on a per-read basis, setting the value to 1 in the configuration file forces all reads to use validation.

During the read from the drive, the content hash is computed. If the hash is wrong, indicating logical drive corruption, the socket is closed before the last block is transmitted, forcing an error to the client.



Note

Using this option creates additional CPU load on the node.

Configuring ROW Replicate On Write

- [ROW versus HP for replication](#)
- [ROW commands](#)
 - [Important](#)
 - [Note](#)
- [Configuration settings](#)
 - [scsp.replicateOnWrite](#)
 - [Important](#)
 - [scsp.defaultROWAction](#)
 - [policy.replication](#)
 - [Tip](#)
 - [Note](#)

Swarm creates all replicas requested at once, in parallel when executing a Replicate On Write (ROW) command for data protection. Objects are safe (replicated on other nodes) using ROW even if a disk failure occurs immediately after a write or update completes.

ROW versus HP for replication

Without ROW, the Health Processor (HP) manages all replication in the background. the HP checks its replication constraint when writing an object. The HP creates a duplicate of the updated version and deletes the older replicas on the cluster nodes when updating an existing object.

With ROW, Swarm creates another instance of the object on another node immediately, as part of the write. ROW verifies two or more object replicas (instances) exist in the cluster before the client write request is completed. If the object includes a constraint for creating more copies, those additional copies are made during the normal health checking process. While ROW may temporarily restrict client responsiveness as objects are created and replicated at the same time, your objects are protected from a single volume failure right away.

ROW benefits	ROW effects
<ul style="list-style-type: none"> • Immediately protects an object from a single-volume failure rather than waiting for the Health Processor to duplicate the object during normal operations. • Quickly deletes older replicas to verify all versions are current. • Quickly invalidates cached domain versions in the cluster so the latest version is implemented. 	<ul style="list-style-type: none"> • Takes a bit longer to write than a single replica. • Is more likely to fail because of the additional preconditions.

ROW commands

ROW is enabled by default. ROW-related commands can be issued in these ways:

Configuration	<p>Disable ROW with this setting in node.cfg. (See Settings Reference.)</p> <pre>scsp.replicateOnWrite = false</pre>
SNMP	<p>Set <code>autoRepOnWrite=0</code> to disable the setting without restarting the cluster. (See Persisted Settings (SNMP).)</p> <div data-bbox="368 562 1485 768" style="border: 1px solid #ccc; padding: 10px;"> <p>Important</p> <p>Once <code>autoRepOnWrite</code> is been changed using SNMP, it is stored in the persistent settings. Any future changes must be using SNMP or the Swarm UI to override that setting.</p> </div>
Query Argument	<p>Use a <code>replicate</code> query argument when needing to override the cluster-wide ROW configuration. (See WRITE with Replicate ROW.)</p> <p>ROW enabled: To override the cluster defaults, use the query argument with an integer. The most common usage, <code>replicate=1</code>, allows the write to succeed with only one instance of the object, which effectively disables ROW for the write:</p> <pre>POST /?replicate=1</pre> <p>ROW disabled: If you have ROW disabled in the cluster, you can achieve ROW by adding the query argument with <code>immediate</code> or <code>full</code> to each write:</p> <pre>POST /?replicate=immediate POST /?replicate=full</pre> <div data-bbox="368 1199 1485 1434" style="border: 1px solid #ccc; padding: 10px;"> <p>Note</p> <p>The <code>replicate=immediate</code> option quickly invalidates cached bucket versions in the cluster so the latest version is implemented in the cluster when creating or updating a bucket. It also prevents subsequent permission errors because out-of-date permissions are used from the prior version.</p> </div>

Configuration settings

Following are the configuration settings that control replication for the storage cluster.

scsp.replicateOnWrite

ROW is enabled by default. You can disable ROW in the storage cluster by changing the setting to `scsp.replicateOnWrite = false`.

The settings `scsp.replicateOnWrite` and `scsp.defaultROWAction` are evaluated together. When `scsp.replicateOnWrite` is true (the default), Swarm withholds sending the write verification to the client until the number of replicas specified by the `scsp.defaultROWAction` setting (default 2) are created successfully.



Important

Any query arguments in the request override the configuration settings.

scsp.defaultROWAction

`scsp.defaultROWAction` is a positive integer or **full** (a special keyword). **Full** indicates that the number of replicas is the number determined by the greater of the object's reps lifepoint or the `policy.replicas default` setting for that object. The limits defined by the cluster `policy.replicas min` and `policy.replicas max` settings are enforced on the number of replicas created in a ROW request as well.

The replicate query argument serves the same purpose with the similar acceptable values.

See [WRITE with Replicate ROW](#).

policy.replication

Your replication policies (`policy.replicas: min:# max:# default:# [anchored]`) and the policy evaluation process affect how Swarm applies ROW. These settings define the baseline range of allowable replicas:

Replication Configured	Effect	Value	Description
<code>scsp.replicateOnWrite</code>	lower limit	true false	Swarm starts evaluation with the value of <code>replicateOnWrite</code> : 1 replica if false and 2 replicas if true (the default).
<code>policy.replicas max</code>	upper limit	$\min \leq n \leq 20$	Swarm determines how many replicas are made for an object and limits the SCSP request to that number. The replication policy <code>default</code> is the key value: if it is 2 and the request asks for more, it receives 2 on the write.

When the `replicate` query argument is used, the remaining `policy.replication` parameters have these effects:

Replication Requested	Meaning	Effect on evaluation	Evaluates to	Description
-----------------------	---------	----------------------	--------------	-------------

replicate=immediate	For use if ROW is disabled in the cluster. Only two replicas must be created before Swarm sends the response; if more replicas are required, those additional replicas are created after the response.	lower limit	2	Only 2 replicas get created synchronously.
replicate=full	For use if ROW is disabled in the cluster. All required replicas must be created before Swarm sends the response.	default value	<code>policy.replicas default</code>	The policy.replicas default evaluated for the object determines the number of replicas Swarm creates synchronously.
replicate={integer}	How many replicas must be created synchronously for this write, overriding cluster settings (within constraints). If policy.replicas max or the lifepoint reps is less than this integer, Swarm uses the smaller of those values. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Tip</p> <p>Use <code>replicate=1</code> to override ROW and achieve the fastest possible writes, for situations that require that.</p> </div>	default limit	<code>{integer} ≤ policy.replicas default</code>	The policy.replicas default evaluated for the object serves as an upper limit on {integer} to determine the number of replicas Swarm creates synchronously.

Note

Neither the replication policy nor the replicate query argument has any effect on erasure-coded objects. Swarm keeps p+1 manifests, up to a limit of `ec.maxManifests` for a k:p encoded EC object.

See [Configuring Cluster Policies](#) and [Implementing Replication Policy](#).

Prometheus Node Exporter and Grafana

- [Hardware Diagnostics with Prometheus](#)
 - [Release history](#)
- [Configuring the Node Exporter](#)
- [Adding Grafana Dashboards](#)
 - [Importing a dashboard](#)
- [Troubleshooting "No Data" Errors](#)
 - [Checking endpoints](#)
 - [Checking Grafana](#)
- [Node Exporter Statistics](#)
 - [Important](#)
 - [Statistics renamed](#)

Hardware Diagnostics with Prometheus

Prometheus is an open-source systems monitoring and alerting toolkit allowing viewing what statistics are available for a system, even under failure conditions.

- [Prometheus](#) scrapes metrics from instrumented jobs, running rules over this data to record aggregated time series or to generate alerts.
- [Grafana](#) and other API consumers can allow visualizing collected data.

The Prometheus Node Exporter is included with Swarm for monitoring and diagnostics on the machines in a Swarm cluster, to provide a wide variety of hardware and kernel related metrics.

i Release history

Storage 11.0 renamed the statistics, so they are globalized for clarity: what used to start with `metrics_` now begins `caringo_swarm_`.

Storage 10.2 added configuration enhancements:

- The service is now enabled by default (`metrics.enableNodeExporter=True`), which makes basic hardware queries across nodes available without reboot.
- The new setting, `metrics.nodeExporterFrequency`, sets how frequently to refresh Swarm-specific metrics; 0 disables this export.

Storage 10.1 introduced the preview of the Prometheus Node Exporter. As a preview, the settings and implementation are subject to change.

- The new setting `metrics.enableNodeExporter` enables Swarm to run the Prometheus node exporter on port 9100.

Configuring the Node Exporter

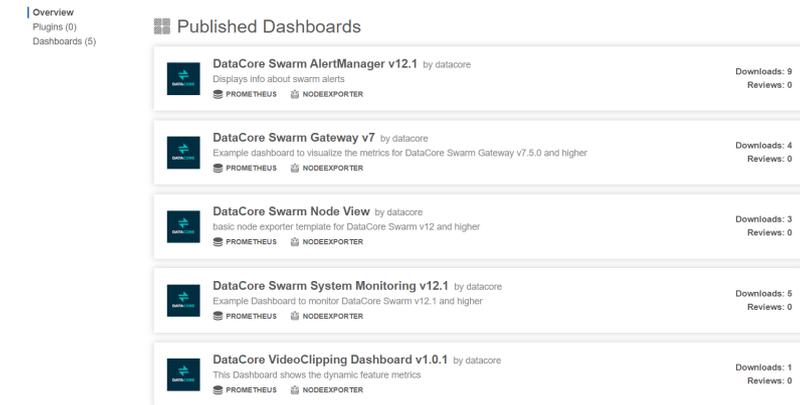
The required Storage setting for Node Exporter is enabled by default: `metrics.enableNodeExporter = True`. A cluster reboot is required to re-enable if disabled.

Change how frequently the exports occur if needed. Perform this using Swarm UI or SNMP on the running cluster: `metrics.nodeExporterFrequency = 120`

Adding Grafana Dashboards

DataCore has published public Grafana dashboards for monitoring Swarm products and features to visualize this Prometheus data. Check here for the latest dashboards for the versions of Swarm products being used:

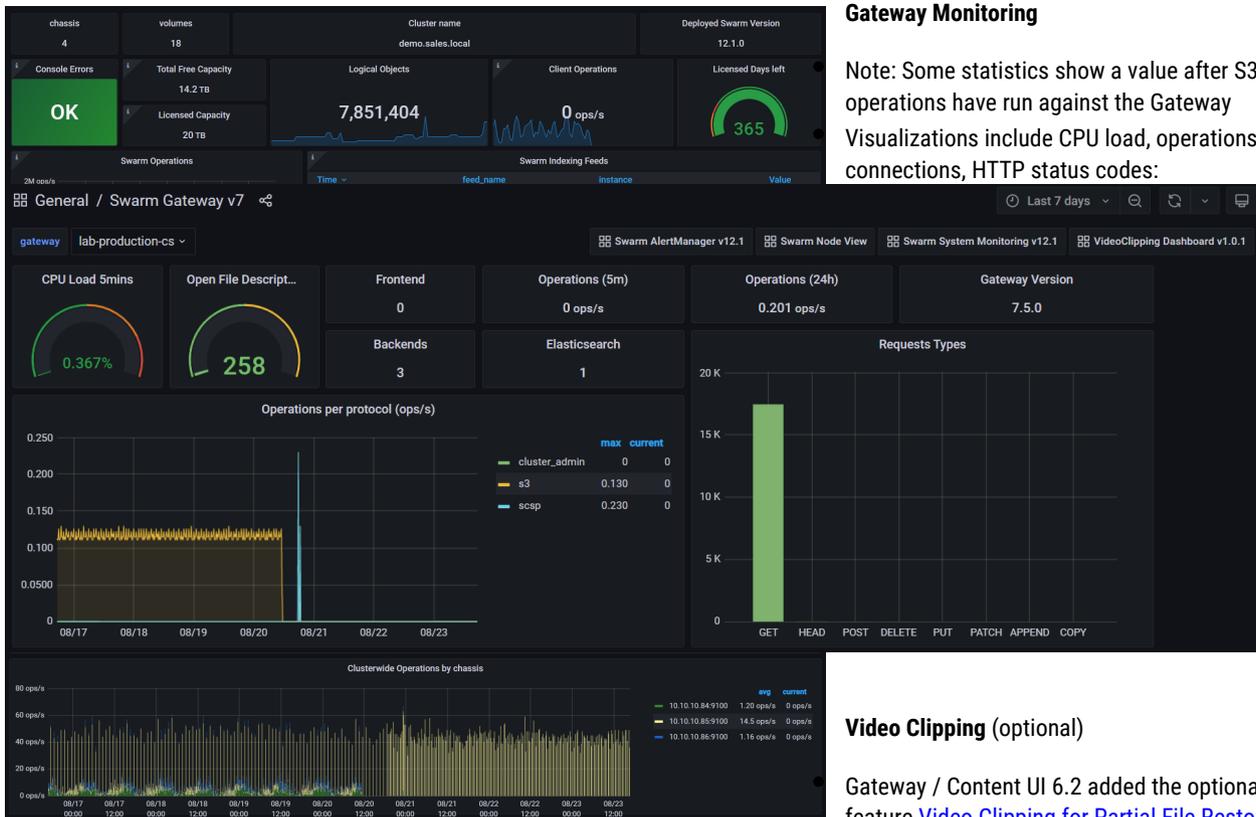
- <https://grafana.com/orgs/datacore>



Customized dashboards are available for the following products:

Swarm System Monitoring (choose the dashboard for the version of Storage)

- Visualizations include cluster health, capacity, indexing, licensing, temperature, and network and CPU loads:
- Cluster-wide operations:
- **Swarm Node View** (new for v12.0)
- Detail view of a single Swarm node:

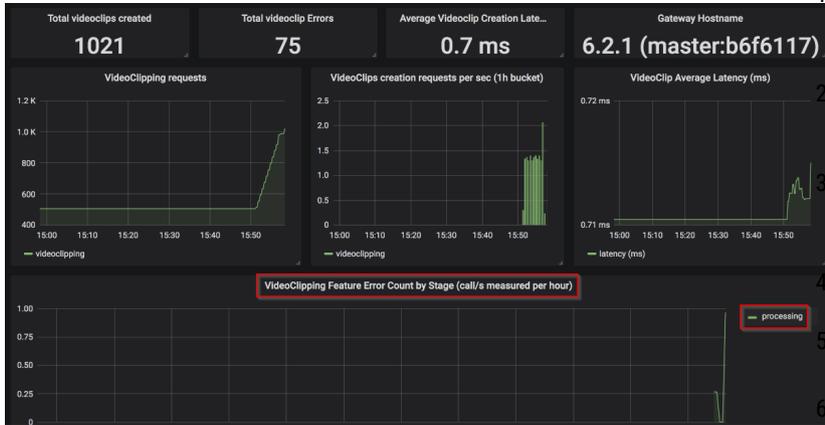


Gateway Monitoring

Note: Some statistics show a value after S3 operations have run against the Gateway
Visualizations include CPU load, operations, connections, HTTP status codes:

Video Clipping (optional)

Gateway / Content UI 6.2 added the optional feature [Video Clipping for Partial File Restore](#).



Grafana Labs Grafana Products Open Source Learn Downloads [Contact us](#) Login

All dashboards » DataCore Swarm System Monitoring v12.1



DataCore Swarm System Monitoring v12.1 by datacore

DASHBOARD

Example Dashboard to monitor DataCore Swarm v12.1 and higher

Last updated: 2 months ago

Start with Grafana Cloud and the new FREE tier. Includes 10K series Prometheus or Graphite Metrics and 50gb Loki Logs

Downloads: 5

Reviews: 0

[Add your review!](#)

Overview Revisions Reviews

Get this dashboard:

14662

[Copy ID to Clipboard](#)

[Download JSON](#)

[How do I import this dashboard?](#)

This dashboard is an example of how to visualize metrics provided by DataCore Swarm v12.1 and above. See DataCore Swarm documentation on instructions for enabling Swarm metrics.

All Swarm metrics are prefixed by the keyword "metrics_"

Visualizations include numbers, rates, and error counts for video clipping requests.

The errors are counted by stage (*preprocessing, processing, postprocessing*), to help with troubleshooting:

Importing a dashboard

- Navigate to <https://grafana.com/get> to obtain a free hosted instance of Grafana (1 user, 5 dashboards).
- View the desired dashboard page and select **Copy ID to Clipboard** to get the ID for the dashboard:
- Open dashboard search on the Grafana instance and then select the **import** button to import a dashboard.
- Paste in the ID when prompted:
- Verify the name is correct once the dashboard is found.
- Important:** Set the **Folder** option to make the dashboard visible. The folder "General" is available by default.
- In the following import process, Grafana prompts setting the data source, and specify any metric prefixes (if the dashboard uses any).

Troubleshooting "No Data" Errors

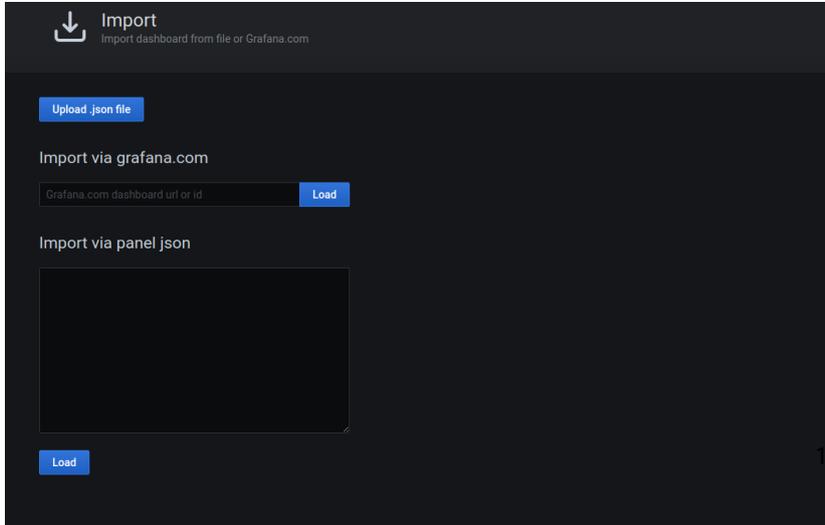
There are multiple points at which things can go wrong in the pipeline from collecting data to displaying charts. The following is the process for troubleshooting No Data errors in graphs.

Checking endpoints

Services monitored by Prometheus (Swarm nodes, Gateways, Elasticsearch) expose an endpoint (usually port 9100), but for Elasticsearch it is `:9200/_prometheus/metrics`.

Swarm – In the Swarm UI Cluster Settings, Advanced, verify `metrics`.
`nodeExporterFrequency=120`.
`metrics.enableNodeExporter=True` must be explicitly set if not on the latest Swarm release. Test the endpoint:

```
curl http://SWARM_NODE:9100/_metr:
```



Elasticsearch – Verify the plugin is installed, permissions are granted, and it was restarted. Test the endpoint:

```
curl http://ELASTICSEARCH:9200/_p
```

Prometheus – Prometheus polls those endpoints as set in `/etc/prometheus/prometheus.ini`. Test the targets:

```
http://PROMETHEUS:9090/targets
```

Checking Grafana

Verify Grafana has a "Prometheus" Data Source and is set to **Default**. The dashboards automatically use this when they are imported; **Edit** a panel to see.

- Verify Grafana is at least version 6.2.1. Some `ldap.toml` configuration features are available in latest releases (6.7.3). Note: older 5.x releases do not display any panels on the Swarm dashboard.

Node Exporter Statistics

Following is information about what Swarm statistics are exported by Prometheus. As possible, these statistics are correlated with MIB entries, although the scales may differ.

i Important

These statistics are reported per node, node-specific feed, or volume, and many need to be aggregated to reflect the cluster status. The statistics including the term "cluster" (in blue) reflect values for the entire cluster.

i Statistics renamed

In Swarm 11, the naming of the statistics has been globalized for clarity: the prefix `metrics_` is now `caringo_swarm_`. (v11.0)

Metric Name (blue indicates cluster-level scope)	Label(s)	Value Meaning	Related SNMP Entry Name(s)
<code>caringo_swarm_cluster_license_capacity_tb</code>	<code>cluster_name</code>	Cluster capacity in terabytes.	<code>totalGBLicensedCapacity</code>
<code>caringo_swarm_cluster_license_days_remaining</code>	<code>cluster_name</code>	Integer number of days remaining on the license.	
<code>caringo_swarm_cluster_license_enabled</code>	<code>cluster_name</code>	1 for license enabled. 0 for not enabled.	
<code>caringo_swarm_cluster_state</code>	<code>cluster_name</code>	-1 = unknown; 0 = ok; 1 = idle; 2 = mounting; 3 = initializing; 4 = finalizing; 5 = maintenance; 6 = retiring; 7 = retired; 8 = error; 9 = unavailable; 10 = offline	<code>clusterState</code>
<code>caringo_swarm_feeds_deleted_pending</code>	<code>feed_name</code> , <code>feed_type</code>	The number of deleted object events pending waiting to be processed.	<code>feedNodeDeletesUnprocessed</code>

caringo_swarm_feeds_deleted_retrying	feed_name, feed_type	The number of deleted object events needing to be retried.	feedNodeDeletesFailing
caringo_swarm_feeds_deleted_successful	feed_name, feed_type	The number of deleted object events successfully processed.	feedNodeDeletesSuccess
caringo_swarm_feeds_deleted_unqualified	feed_name, feed_type	The number of deleted object events potentially requiring processing.	feedNodeDeletesUnqualified
caringo_swarm_feeds_est_backlog_clear_time	feed_name, feed_type	The estimated number of seconds to complete all processing. -1 for unknown.	feedEstBacklogClearTime
caringo_swarm_feeds_existing_pending	feed_name, feed_type	The number of current object events pending waiting to be processed.	feedNodeExistsUnprocessed
caringo_swarm_feeds_existing_retrying	feed_name, feed_type	The number of current object events needing to be retried.	feedNodeExistsFailing
caringo_swarm_feeds_existing_successful	feed_name, feed_type	The number of current object events successfully processed.	feedNodeExistsSuccess
caringo_swarm_feeds_existing_unqualified	feed_name, feed_type	The number of current object events potentially requiring processing.	feedNodeExistsUnqualified
caringo_swarm_feeds_feed_id	feed_name, feed_type	The id number of the feed.	feedFeedId
caringo_swarm_feeds_feed_state	feed_name, feed_type	-1 = unknown; 0 = closed; 1 = config-error; 2 = too many overlapping feeds; 3 = blocked; 4 = paused by request; 5 = paused for recovery; 6 = priority (processing contexts after start /restart); 7 = ok	feedState
caringo_swarm_feeds_last_failure	feed_name, feed_type	The time of the last failure event in epoch milliseconds.	feedLastExistFailure, feedLastDeleteFailure, feedLastVersionedFailure
caringo_swarm_feeds_last_success	feed_name, feed_type	The time of the last successful event in epoch milliseconds.	feedLastSuccess
caringo_swarm_feeds_remote_failure	feed_name, feed_type	The number of replication/indexing failures.	feedPluginRemoteFailure
caringo_swarm_feeds_remote_success_duplicate	feed_name, feed_type	The number of duplicated indexing /replication successes.	feedPluginRemoteSuccessDuplicate
caringo_swarm_feeds_remote_success_transfer	feed_name, feed_type	The number of new indexing/replication successes.	feedPluginRemoteSuccessTransfer
caringo_swarm_feeds_versioned_pending	feed_name, feed_type	The number of versioned object events pending waiting to be processed.	feedNodeVersionedUnprocessed
caringo_swarm_feeds_versioned_retrying	feed_name, feed_type	The number of versioned object events needing to be retried.	feedNodeVersionedFailing
caringo_swarm_feeds_versioned_successful	feed_name, feed_type	The number of versioned object events successfully processed.	feedNodeVersionedSuccess
caringo_swarm_feeds_versioned_unqualified	feed_name, feed_type	The number of versioned object events potentially requiring processing.	feedNodeVersionedUnqualified
caringo_swarm_health_cycle		The HP cycle number.	ongoingHPCycleNumber

caringo_swarm_health_examined		The number of streams examined so far this HP cycle.	ongoingHPCycleStreamsExamined
caringo_swarm_health_offloaded		The number of streams moved to another node this HP cycle.	ongoingHPCycleStreamsOffloaded
caringo_swarm_health_relocated		The number of streams relocated on disk this HP cycle.	ongoingHPCycleStreamsRelocated
caringo_swarm_health_total		The number of streams processed so far this HP cycle.	ongoingHPCycleStreamsTotal
caringo_swarm_health_verified		The number of streams checked for data integrity this HP cycle.	ongoingHPCycleStreamsVerified
caringo_swarm_index_alias_slots		The number of memory index slots used for alias objects.	indexSlotsAlias
caringo_swarm_index_deleted_slots		The number of memory index slots used for deleted objects.	indexSlotsDeleted
caringo_swarm_index_immutable_slots		The number of memory index slots used for immutable objects.	indexSlotsImmutable
caringo_swarm_index_manifest_slots		Not useful. Always 0.	indexSlotsManifest
caringo_swarm_index_mutable_slots		The number of memory index slots used for named+alias objects.	indexSlotsMutable
caringo_swarm_index_named_slots		The number of memory index slots used for named objects.	indexSlotsNamed
caringo_swarm_index_overlay_slots		The number of memory index slots used for the overlay index.	indexSlotsOverlayUsed
caringo_swarm_index_policy_slots		The number of memory index slots used for policy attributes.	indexSlotsPolicy
caringo_swarm_index_total_slots		The number of memory index slots total.	indexSlotsTotal
caringo_swarm_index_used_slots		The number of memory index slots used.	indexSlotsUsed
caringo_swarm_index_versioned_slots		The number of memory index slots used for prior object versions.	indexSlotsVersioned
caringo_swarm_memory_cache_memory_allocated		The memory allocated to the content cache in bytes.	contentCacheCapacityMB
caringo_swarm_memory_cache_memory_items		The number of objects stored in the content cache.	contentCacheItems
caringo_swarm_memory_cache_memory_used		The memory used to store objects in the content cache in bytes.	contentCacheUsedMB
caringo_swarm_memory_chassis_arena		The bytes of memory available for use by Swarm on the chassis.	chassisArenaM
caringo_swarm_memory_chassis_free		The bytes of memory free on the chassis.	chassisFreeMemM
caringo_swarm_memory_chassis_headroom		The bytes of memory reserved for emergency use on the chassis.	chassisHeadroomM
caringo_swarm_memory_chassis_shared		The bytes of shared memory used on the chassis.	chassisSharedMemM

caringo_swarm_memory_chassis_total		The bytes of physical memory on the chassis.	chassisTotalMemM
caringo_swarm_memory_node_accounted		Bytes of buffer memory in use in the main process.	accountedMemM
caringo_swarm_memory_node_accounts		Number of memory accounts in used in the main process.	memAccountsActual
caringo_swarm_memory_node_accounts_over_budget		Number of memory accounts over budget in the main process.	memAccountsOverlimit
caringo_swarm_memory_node_accounts_throttled		Number of memory accounts throttled in the main process.	memAccountsQueued
caringo_swarm_memory_node_actual		Total bytes in use for the main process.	processActualSizeM
caringo_swarm_memory_node_allowance		Total buffer memory allocated in the main process.	accountAllowanceM
caringo_swarm_memory_node_file_descriptors		Number of file descriptors in use by the main process.	processFDs
caringo_swarm_memory_node_non_accounted		Bytes of non accounted memory in use in the main process.	nonAccountedMemM
caringo_swarm_memory_node_target		Main process target size in bytes.	processTargetSizeM
caringo_swarm_node_errors		The number of reported errors on the node.	castorErrTableSize
caringo_swarm_node_examq		The number of examination queue entries on the node.	examQueueCount
caringo_swarm_node_state		-1 = unknown; 0 = ok; 1 = idle; 2 = mounting; 3 = initializing; 4 = finalizing; 5 = maintenance; 6 = retiring; 7 = retired; 8 = error; 9 = unavailable; 10 = offline	castorState
caringo_swarm_node_swarm_version	version	Value is always 1.	castorVersion
caringo_swarm_node_uptime		The uptime of the main process in seconds.	sysUpTimeInstance
caringo_swarm_node_volumes		The number of volumes in use on the node.	castorVolumes
caringo_swarm_scsp_appends caringo_swarm_scsp_appends_total		The delta since last publication or the total number of APPEND requests.	appends
caringo_swarm_scsp_client_close_read caringo_swarm_scsp_client_close_read_total		The delta since last publication or the total number of client premature closes on read-type requests.	clientPrematureCloseRead
caringo_swarm_scsp_client_close_write caringo_swarm_scsp_client_close_write_total		The delta since last publication or the total number of client premature closes on write-type requests.	clientPrematureCloseWrite
caringo_swarm_scsp_copies caringo_swarm_scsp_copies_total		The delta since last publication or the total number of COPY requests.	copies
caringo_swarm_scsp_deletes caringo_swarm_scsp_deletes_total		The delta since last publication or the total number of DELETE requests.	deletes

caringo_swarm_scsp_gets caringo_swarm_scsp_gets_total		The delta since last publication or the total number of GET requests.	reads
caringo_swarm_scsp_heads caringo_swarm_scsp_heads_total		The delta since last publication or the total number of HEAD requests.	infos
caringo_swarm_scsp_indirectDeletes caringo_swarm_scsp_indirectDeletes_total		The delta since last publication or the total number of deletes performed internally by the health processor.	indirectDeletes
caringo_swarm_scsp_internode_reads caringo_swarm_scsp_internode_reads_total		The delta since last publication or the total number of GET requests internally performed.	internodeReads
caringo_swarm_scsp_internode_redirects caringo_swarm_scsp_internode_redirects_total		The delta since last publication or the total number of client redirects between nodes in the cluster.	redirects
caringo_swarm_scsp_internode_trims caringo_swarm_scsp_internode_trims_total		The delta since last publication or the total number of replicas internally removed.	internodeTrims
caringo_swarm_scsp_internode_writes caringo_swarm_scsp_internode_writes_total		The delta since last publication or the total number of POST requests internally performed.	internodeWrites
caringo_swarm_scsp_patches caringo_swarm_scsp_patches_total		The delta since last publication or the total number of PATCH requests.	patches
caringo_swarm_scsp_posts caringo_swarm_scsp_posts_total		The delta since last publication or the total number of POST requests.	writes
caringo_swarm_scsp_puts caringo_swarm_scsp_puts_total		The delta since last publication or the total number of PUT requests.	updates
caringo_swarm_scsp_response_200 caringo_swarm_scsp_response_200_total		The delta since last publication or the total number of SCSP 200 responses.	clientSuccess200
caringo_swarm_scsp_response_201 caringo_swarm_scsp_response_201_total		The delta since last publication or the total number of SCSP 201 responses.	clientSuccess201
caringo_swarm_scsp_response_202 caringo_swarm_scsp_response_202_total		The delta since last publication or the total number of SCSP 202 responses.	clientSuccess202
caringo_swarm_scsp_response_206 caringo_swarm_scsp_response_206_total		The delta since last publication or the total number of SCSP 206 responses.	clientSuccess206
caringo_swarm_scsp_response_301 caringo_swarm_scsp_response_301_total		The delta since last publication or the total number of 301 redirect responses.	clientRedir301
caringo_swarm_scsp_response_304 caringo_swarm_scsp_response_304_total		The delta since last publication or the total number of 304 redirect responses.	clientRedir304
caringo_swarm_scsp_response_400 caringo_swarm_scsp_response_400_total		The delta since last publication or the total number of 400 error responses.	clientError400
caringo_swarm_scsp_response_401 caringo_swarm_scsp_response_401_total		The delta since last publication or the total number of 401 error responses.	clientError401
caringo_swarm_scsp_response_404 caringo_swarm_scsp_response_404_total		The delta since last publication or the total number of 404 error responses.	clientError404
caringo_swarm_scsp_response_410 caringo_swarm_scsp_response_410_total		The delta since last publication or the total number of 410 error responses.	clientError410

caringo_swarm_scsp_response_412 caringo_swarm_scsp_response_412_total		The delta since last publication or the total number of 412 error responses.	clientError412
caringo_swarm_scsp_response_4xx caringo_swarm_scsp_response_4xx_total		The delta since last publication or the total number of other 400-type error responses.	clientError4xx
caringo_swarm_scsp_response_500 caringo_swarm_scsp_response_500_total		The delta since last publication or the total number of 500 error responses.	clientError500
caringo_swarm_scsp_response_503 caringo_swarm_scsp_response_503_total		The delta since last publication or the total number of 503 error responses.	clientError503
caringo_swarm_scsp_response_507 caringo_swarm_scsp_response_507_total		The delta since last publication or the total number of 507 error responses.	clientError507
caringo_swarm_scsp_response_5xx caringo_swarm_scsp_response_5xx_total		The delta since last publication or the total number of other 500-type error responses.	clientError5xx
caringo_swarm_scsp_searches caringo_swarm_scsp_searches_total		The delta since last publication or the total number of search requests.	searches
caringo_swarm_volume_capacity	volume_dev, volume_id	The volume capacity in bytes.	volMaxMbytes
caringo_swarm_volume_ecrs	volume_dev, volume_id	The number of EC recoveries ongoing against this volume.	recoveryType, recoveryLocalVolId
caringo_swarm_volume_errors	volume_dev, volume_id	The number of reported IO errors on the volume.	volErrors
caringo_swarm_volume_free	volume_dev, volume_id	The number of free bytes on the volume.	volFreeMbytes
caringo_swarm_volume_fvrs	volume_dev, volume_id	The number of failed volume recoveries ongoing against this volume.	recoveryType, recoveryLocalVolId
caringo_swarm_volume_journal_utilization	volume_dev, volume_id	The portion of the volume journal space in use.	volLastJournalBid
caringo_swarm_volume_logical_objects	volume_dev, volume_id	The contribution to estimated cluster logical objects from this volume.	logicalObjects
caringo_swarm_volume_logical_space	volume_dev, volume_id	The contribution to estimated cluster logical space (in bytes) from this volume.	logicalSpace
caringo_swarm_volume_logical_unprocessed	volume_dev, volume_id	The number of streams on the volume not considered for the logical object /space estimates.	logicalUnprocessed
caringo_swarm_volume_read_bid	volume_dev, volume_id	The last read bid for the volume.	lastRead
caringo_swarm_volume_rep_bid	volume_dev, volume_id	The last replicate bid for the volume.	lastWrite
caringo_swarm_volume_state	volume_dev, volume_id	The status of the given volume name and ID. Statuses: 0 (OK), 1 (retiring), 2 (retired), 3 (unavailable), 4 (mounting), 5 (idle), -1 (unknown).	volState

caringo_swarm_volume_stats_io_queue_count	volume_dev, volume_id	The number of IO queue items on the last sampling.	
caringo_swarm_volume_stats_io_queue_sec	volume_dev, volume_id	The time in seconds to process items on the IO queue at the last sampling.	
caringo_swarm_volume_stats_io_utilization	volume_dev, volume_id	The fraction of the time at the last sampling the volume was busy.	
caringo_swarm_volume_stats_sec_per_io_max	volume_dev, volume_id	The longest IO request time at the last sampling.	
caringo_swarm_volume_stats_sec_per_io_running	volume_dev, volume_id	The average IO request time at the last sampling.	
caringo_swarm_volume_streams	volume_dev, volume_id	The number of streams on the volume.	volUsedstreams
caringo_swarm_volume_trapped	volume_dev, volume_id	The trapped space on the volume in bytes.	volTrappedMbytes
caringo_swarm_volume_uptime	volume_dev, volume_id	The time in seconds the volume has been up.	volUptime
caringo_swarm_volume_used	volume_dev, volume_id	The number of bytes used on the volume.	volUsedMbytes
caringo_swarm_volume_write_bid	volume_dev, volume_id	The last write bid for the volume.	

Exporting Prometheus Data

- [Tip](#)
 - [ExportToCsv.py](#)
 - [Results](#)

If Prometheus data needs to be exported (such as to supply to machine learning), adapt the Python 2 script below to run a Prometheus 2.x query that outputs to a CSV file.

Tip

By default, Prometheus retains your metrics data for 14 days (which is configurable). Use this script in a cron job to collect and archive all historical data.

ExportToCsv.py

```
#!/usr/bin/env python

import csv
import requests
import sys

if len(sys.argv) != 3:
    print('Usage: {0} http://prometheus:9090 a_query'.format(sys.argv[0]))
    sys.exit(1)

response = requests.get('{0}/api/v1/query'.format(sys.argv[1]),
    params={'query': sys.argv[2]})
results = response.json()['data']['result']

labelnames = set()
for result in results:
    labelnames.update(result['metric'].keys())

labelnames.discard('__name__')
labelnames = sorted(labelnames)

writer = csv.writer(sys.stdout)
writer.writerow(['name', 'timestamp', 'value'] + labelnames)

for result in results:
    for avalue in result['values']:
        l = [result['metric'].get('__name__', '')] + avalue
        for label in labelnames:
            l.append(result['metric'].get(label, ''))
        writer.writerow(l)
```

The script expects two arguments:

- the URL to the Prometheus endpoint, which defaults to `http://127.0.0.1:9090`
- a PromQL (native Prometheus) query
 - See the reference for the PromQL query language: <https://prometheus.io/docs/prometheus/latest/querying/examples/>

This is the output that the above script generates:

Results

```
[root@swarmtelemetry ~]# ./ExportToCsv.py http://127.0.0.1:9090 caringo_swarm_scsp_gets[5m]
name,timestamp,value,instance,job
caringo_swarm_scsp_gets,1584979640.11,382,10.10.10.84:9100,swarm
caringo_swarm_scsp_gets,1584979670.11,382,10.10.10.84:9100,swarm
caringo_swarm_scsp_gets,1584979700.11,382,10.10.10.84:9100,swarm
caringo_swarm_scsp_gets,1584979730.11,294,10.10.10.84:9100,swarm
caringo_swarm_scsp_gets,1584979760.11,234,10.10.10.85:9100,swarm
caringo_swarm_scsp_gets,1584979790.11,284,10.10.10.85:9100,swarm
caringo_swarm_scsp_gets,1584979820.11,294,10.10.10.85:9100,swarm
```

Swarm Storage Policies

Design and apply precise policies for where and how to apply protections in the implementation to make the fullest use of the rich content protection features of Swarm. **Policies** define how the data being uploaded to the Swarm cluster is stored, managed, and protected.

How it works

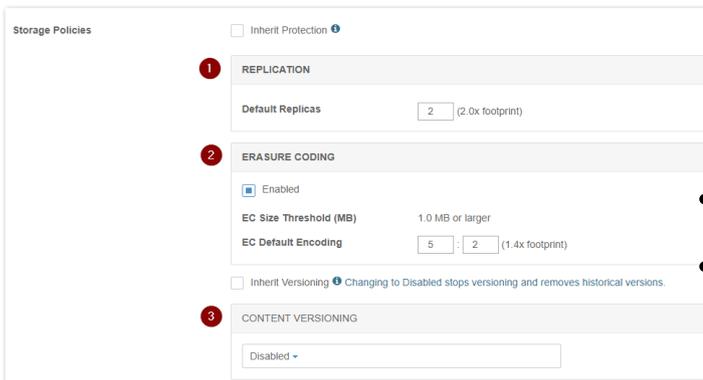
Cluster-level policies reside in configuration settings and allows defining cluster-wide and cluster-specific policies for how Swarm implements the content protection features (see [Configuring Cluster Policies](#)). These baseline cluster policies can be layered with *context-level* (domain and bucket) policies, which reside in designated headers in the context objects themselves. Policies are not [lifepoints](#): they do not have built-in lifetimes, nor do they specify transitions from one policy to another over time. Policies are only changed by explicit updates.

Types of content policies

Policies can be set for these content protection options, specific to the context needed:

1. [Replication](#) - how many default, minimum, and maximum number of replicas to keep for the objects in this cluster/domain/bucket
2. [Erasure Coding EC](#) - whether to enable or specify the EC encoding ($k:p$) to use for the objects in this cluster/domain/bucket
3. [Object Versioning](#) - whether to enable, disable, or suspend versioning for the objects in this cluster/domain/bucket

These appear under the bucket or domain's **Properties** (gear) tab, **Storage Policies** section in the Content UI:



How policies resolve

Swarm begins policy evaluation by eliminating contexts (domain and bucket) not applicable to resolving overlapping policies for one of two reasons:

- because of the "anchored" parameter, which overrides lower-level policies
- because of the type of object, which restricts it to certain contexts:

Swarm (1) selects the anchored policy if it exists, or else (2) checks the relevant contexts from the bottom up and selects the first policy it finds.

i

Note

Within a given request to a context (domain or bucket) object, if more than one policy header of the same type is written, the last one written is honored.

Setting policies by cluster

Cluster-specific policies for domains and buckets can be set by including the cluster name parenthetically and separating them with a comma. Swarm looks for and takes any cluster-specific policy values, using the default value only if it finds no match.

A policy (such as versioning) may be desired to be enabled in the main cluster but disabled in the target of the replication feed. This is how a feature is enabled in the main cluster (myCluster) but disabled in the one-way replication cluster (myRemote):

```
Policy-{feature}: enabled (myCluster), disabled (myRemote), disabled
or
Policy-{feature}: enabled (myCluster), disabled
```

Required

Always end a multi-part policy with the preferred default value.

Updating policies

Swarm cluster policies are all [persisted settings](#), so use the **snmpset** command on the cluster's settings object to change and persist the policies cluster-wide. Domain and bucket policies are saved as headers on those objects.

See [Using SNMP with Swarm](#).

Reducing Redundancy for Lesser Content

To make the most cost-effective use of your storage footprint, you can reduce the redundancy of Swarm's content protection. These are types of content that may be candidates for reduced redundancy:

- Nightly backups, which you can restore from using any save set within a few days
- Intermediate files in a multi-step operation
- Files that can be recreated on demand, as needed
- Content of little business impact if lost

Balancing the downside to losing the file versus the likelihood that particular one is needed, protection can be lowered as described below.

How to enable reduced redundancy

1. First, lower the cluster-wide minimum for replication, which sets the absolute lower bound for the cluster: Change `policy.replicas` to `min:1`, if it is a 2 or higher.

Policy		
policy.eCEncoding	5:2	Default: unspecified anchored
policy.eCMinStreamSize	1Mb	Default: 1Mb anchored
policy.replicas	min:1 max:16 default:2 anchored	Default: min:2 max:16 default:2 anchored
policy.versioning	allowed	Default: disallowed

```
policy.replicas min:1 max:<int> default:<int>
```

Note: the *minimum* is not the *default*. This setting can be changed dynamically, using the UI or SNMP. See [Implementing Replication Policy](#).

Caution

Parity of 1 for erasure coding lowers protection and so should *not* be used as your cluster's default (`policy.ecEncoding=N:1`); as N increases, N:1 approaches the protection of having only a single copy of the object in the cluster.

2. *Best practice:* If a category of lesser content regularly needs to be store, provide it its own container (context) and set reduced redundancy on that container only.
 - a. Create or designate a specific domain or bucket to be dedicated to this content.
 - b. Update the domain or bucket properties to enforce single replicas, no erasure coding, and no versioning, using the Content UI or a manual command:

The screenshot shows the 'Storage Policies' configuration page. Under the 'REPLICATION' section, 'Default Replicas' is set to 1. Under the 'ERASURE CODING' section, the 'Enabled' checkbox is unchecked. There are also 'Inherit Protection' and 'Inherit Versioning' options at the top and bottom of the policy configuration area.

```
curl -iL -XPOST --post301 --data-binary ""
-H "Policy-Replicas: default:1"
-H "Policy-ECEncoding: disabled"
-H "Policy-Versioning: disabled"
"http://{cluster}/myBucket?domain=myDo"
```

See [Implementing EC Encoding Policy](#).

Why disable erasure coding?

When storing a category of lesser content with reduced redundancy, erasure coding should not be used. If an erasure-coded object degrades (loses more than p segments in a $k:p$ encoding, such as 2:1), then the remaining segments still take up cluster space, and the health processor continues to look for k segments to attempt to reconstruct missing ones. With whole replica encoding, losing lesser data results in the entire object being removed from the cluster, reclaiming that space and avoiding extra processing.

3. Apart from your special domain or bucket, when you have an individual file (such as a backup) that is a candidate for reduced redundancy, write it with lifepoint headers that lower redundancy. See [Lifepoint Metadata Headers](#). In this example, a chunked upload (which *must* be EC encoded) has two lifepoints: the first lifepoint specifies an EC encoding that expires in one day, and the second specifies that the cluster needs to keep only one replica after that.

```
Transfer-Encoding: chunked
Lifepoint: [Wed, 4 Apr 2019 15:59:02 GMT] reps=2:1
Lifepoint: [] reps=1
```

Object Versioning

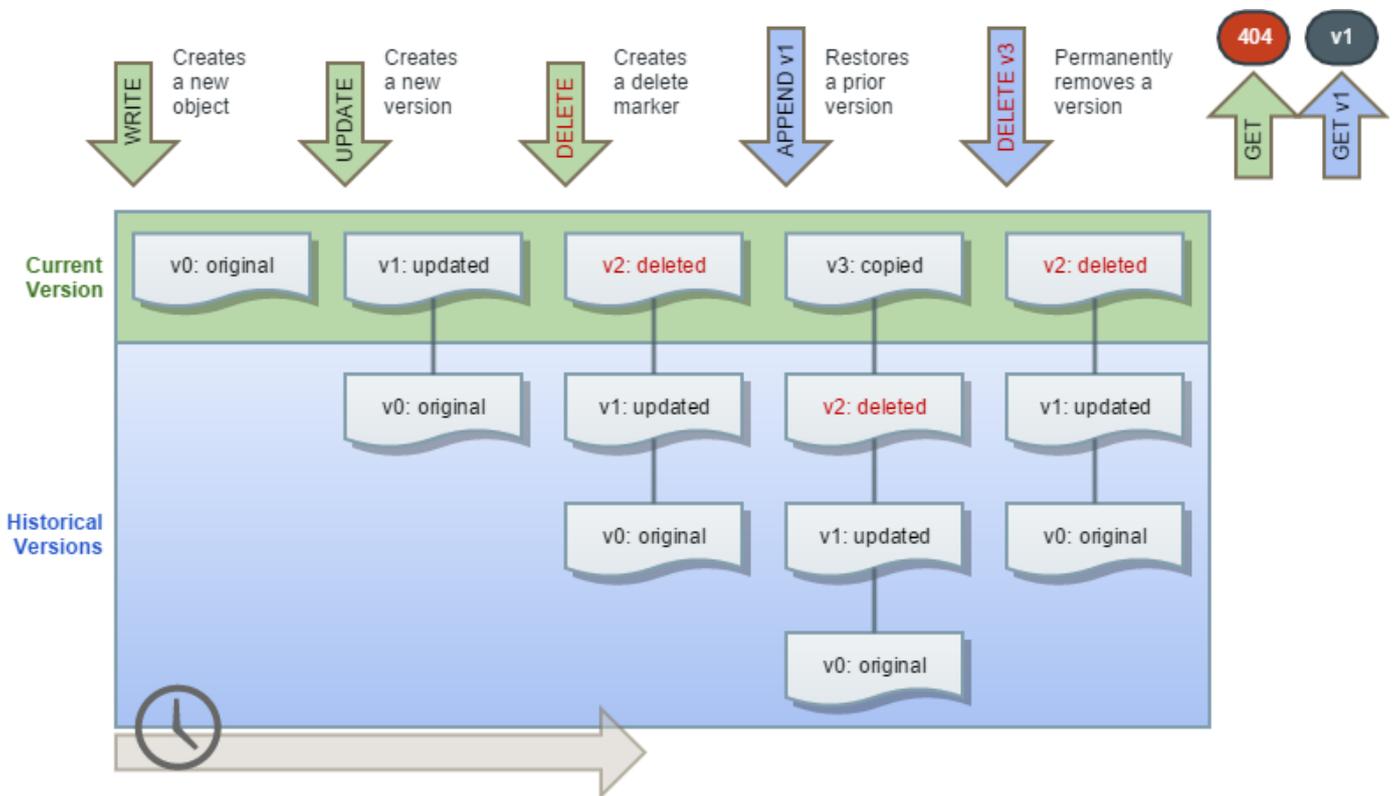
- [Note](#)
- [S3 Versioning](#)
- [Why use versioning?](#)
- [What is versioned?](#)

Object-level versioning is a powerful content protection option that tracks, secures, and provides access to historical versions of objects, even after they are deleted. With versioning, applications can read, list, revert, and purge prior versions as well as restore objects deleted by mistake.

Note

Using Swarm versioning with SCSP operations has no dependencies. To use Swarm versioning with [Amazon S3](#), Content Gateway version 4.1 or higher must be run.

Versioning preserves a set of historical variants of an object, the original plus subsequent updates to it, up to and including the latest version:



These are key capabilities of Swarm versioning:

- **Unlimited versions** – The number of supported versions for a given object is unbounded, and all versions have a unique version ID. List all versions, access, restore, and permanently delete specific versions via the version ID.
- **Flexible policy** – The cluster administrator changes the cluster policy settings to allow versioning; the domain administrator can then allow and even require versioning in that domain. A bucket owner can enable/disable versioning for a specific bucket if allowed by the cluster and domain.

- **Lossless concurrent updates** – Swarm captures simultaneous PUT updates and resolves the order in the version chain. Swarm preserves all versions, even those overlapping in time, with the latest update as the current version.
- **Accurate disk reporting** – Each object revision in a domain/bucket with versioning-enabled preserves and reports the full size on disk. Swarm includes all object revisions in the 'du' responses if requested. The size for deleted and historical versions counts towards bucket and domain totals.
- **Support for search and replication** – Swarm Versioning works with both [Search feeds](#) and [Replication feeds](#), provided all clusters are running the same version of Swarm.

S3 Versioning

Swarm's native object versioning feature is interoperable with AWS S3 versioning. The implementation includes these improvements:

- *Ability to disable versioning:*
AWS S3 allows for versioning to be suspended once enabled on a bucket. Swarm provides the ability to disable versioning and automatically clean up the prior versions to reclaim storage space.
- *Delete marker consolidation:*
Unlike AWS S3 where continued DELETE operations on a deleted object record additional delete markers in the version history, Swarm acknowledges the subsequent deletes without recording additional delete markers. Multi-factor authentication delete is not supported.
- *Expanded version listing:*
Swarm supports version listing batches up to 2000 items while AWS S3 limits these listing results to batches of 1000. Additionally, Swarm does not break batches on version boundaries. Delimiter case is currently not supported for version listing.
- *Simplified ACL management:*
When using per-object ACLs with versioning, the ACL for the current version of the object applies for determining authorization. To change the ACL for an object's entire version chain, update the object *without* specifying a version.

Why use versioning?

Versioning meets two key needs:

- Require extremely durable data retention and archiving.
- Require the ability to recover when data is erroneously overwritten or deleted.

With versioning enabled, prior versions of a stored object, can be retrieved and restored allowing recovery from data loss, whether caused by user error or application failure:

- **Deleting an object** – Swarm inserts a delete marker instead of removing it permanently, which becomes the current object version. Previous versions can be restored.
- **Overwriting an object** – Swarm performs the update by creating a new version, allowing restoring previous versions by rolling back a bad update.

By default, versioning is disabled across the cluster. To avoid excessive storage usage, enable versioning in a targeted way, where change control is required.

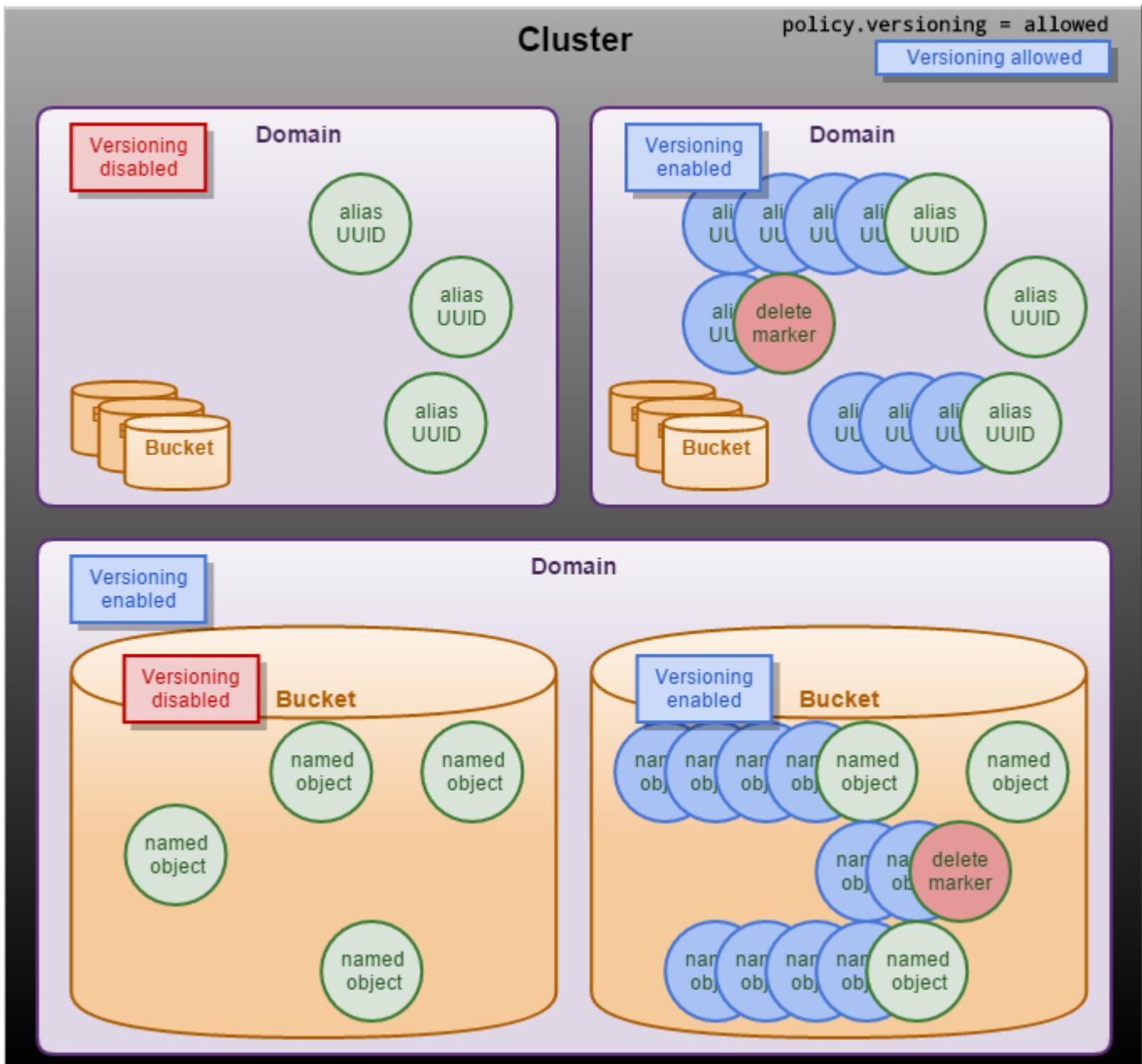
What is versioned?

Choosing to use versioning enables the ability to preserve, retrieve, and restore every update of every object stored in that context (domain or bucket). With Versioning, Swarm archives another copy of an existing object when an update or delete is processed. GET requests retrieve the most recently written version, but retrieval of older versions of an object is performed by specifying a version in the request.

Administrators can selectively enable versioning at the following levels once the **cluster** is configured to allow versioning:

- the **domain** level (for *alias* objects)

- the **bucket** level (for *named* objects)



When a DELETE operation is issued against a versioned object, Swarm creates a delete marker so subsequent unversioned requests no longer retrieve the object. Swarm stores all versions of an object so they can be retrieved and restored.

Note: these types of Swarm objects cannot be versioned:

- Domains
- Buckets
- Unnamed objects (which are immutable)
- Alias objects not tenanted in a domain

Objects are versioned while domains and buckets are not (contexts). A bucket *is* lost if accidentally deleted. Swarm pauses the recursive delete of the bucket's contents for the duration of the grace period ([health.recursiveDeleteDelay](#)). There is time to recreate the bucket with the same headers to avoid data loss (see [Restoring Domains and Buckets](#)). The content starts to disappear as Swarm's Health Processor begins cleaning up all versions of the obsolete content, to reclaim space, if the bucket is not restored and the grace period expires.

- [Versioning Examples](#)
- [Working with Versioning](#)
- [Implementing Versioning](#)
- [Versioning Operations](#)

Versioning Examples

These examples show how to use cURL commands for working with versioning.

- [Enable versioning on a cluster](#)
- [Enable versioning on a domain](#)
 - [Enable versioning on an existing domain via Swarm](#)
 - [Enable versioning on an existing domain via Gateway](#)
- [Enable versioning on a bucket](#)
- [Create a named object in the bucket](#)
- [List versions in the bucket](#)
 - [Tip](#)
- [Update the named object](#)
- [View the new version in the listing](#)
- [INFO the prior version](#)
- [Delete the prior version](#)
- [Delete the current version](#)
- [Check the delete marker](#)

Enable versioning on a cluster

Use an SNMP set command to allow versioning in a cluster, adjusted for the environment:

```
snmpset -m +CARINGO-CASTOR-MIB -v2c -M +/usr/share/snmp/mib2c-data -cPASSWORD -Oqs {cluster} clus
```

Enable versioning on a domain

Set the `Policy-Versioning` header and include the `admin` query argument to enable versioning for an existing domain:

Enable versioning on an existing domain via Swarm

```
curl --anyauth -u admin:password -i --location-trusted -X PUT --post301 --data-binary ''
-H "Policy-Versioning: enabled" "{cluster}/?domain=sample&replicate=immediate&admin"
```

```
HTTP/1.1 201 Created
Location: {cluster}/?domain=sample
Volume: ec8ab948651deb7b9599d2288cf349e6
Location: {cluster}/?domain=sample
Volume: ae4adcf66f67f85ceb8096585fe8aa5e
Castor-System-Owner: @CAStor administrator
Entity-MD5: g9WWPqq3KoKxB5+dko3Taw==
Stored-Digest: 83d5963eaab72a82b1079f9d928dd36b
Last-Modified: Fri, 26 Jun 2015 21:52:07 GMT
Content-UUID: 157b02492dfb48d86fad8a0935ba440a
Castor-System-Version: 143535527.814
Etag: "7f6d0be2bd765c68f41a960e8035e0f6"
Castor-System-Alias: 157b02492dfb48d86fad8a0935ba440a
Replica-Count: 2
Date: Fri, 26 Jun 2015 21:52:08 GMT
Server: CAStor Cluster/7.5.1
Content-Length: 46
Content-Type: text/html
Keep-Alive: timeout=14400
```

Enable versioning on an existing domain via Gateway

```
curl --anyauth -u cs3admin:password -i --location-trusted -X PUT --post301 --data-binary ''
-H "Policy-Versioning: enabled" "{cluster}/?domain=sample&replicate=immediate&admin"
-v --anyauth
-u cs3admin:caringo
-H 'Content-Type: application/castorcontext'
```

Review the headers returned to verify the versioning state for a domain:

```
curl -i --location-trusted --anyauth -u admin:ourpwdofchoicehere
"{cluster}/?domain=sample&admin"

HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest realm="CASTor administrator", nonce="3962288e3bb66797270d5a40d0d22468",
  opaque="f30a386668dae501437669a5572f12fe", stale=false, qop="auth", algorithm=MD5
WWW-Authenticate: Basic realm="CASTor administrator"
Content-Length: 53
Content-Type: text/html
Date: Fri, 26 Jun 2015 21:52:22 GMT
Server: CASTor Cluster/7.5.1
Allow: HEAD, HOLD, GET, SEND, PUT, RELEASE, POST, COPY, GEN, APPEND, DELETE
Keep-Alive: timeout=14400

HTTP/1.1 200 OK
Castor-System-Alias: 157b02492dfb48d86fad8a0935ba440a
Castor-System-CID: ffffffffffffffffffffffffffffffffff
Castor-System-Cluster: Sample
Castor-System-Created: Fri, 26 Jun 2015 21:52:07 GMT
Castor-System-Name: sample
Castor-System-Owner: @CASTor administrator
Castor-System-Version: 143535527.814
Content-Length: 0
Content-Type: application/x-www-form-urlencoded
Last-Modified: Fri, 26 Jun 2015 21:52:07 GMT
Policy-Versioning: enabled
Etag: "7f6d0be2bd765c68f41a960e8035e0f6"
Castor-System-Path: /sample
Castor-System-Domain: sample
Volume: ae4adcf66f67f85ceb8096585fe8aa5e
Volume-Hint: ec8ab948651deb7b9599d2288cf349e6
Feed-0-Status: 0
Feed-0-StatusTime: Fri, 26 Jun 2015 21:52:09 GMT
Policy-Versioning-Evaluated: enabled
Date: Fri, 26 Jun 2015 21:52:22 GMT
Server: CASTor Cluster/7.5.1
Keep-Alive: timeout=14400
```

Notice `Policy-Version-Evaluated` is enabled. *All tenanted alias objects* in the domain are versioned.

Enable versioning on a bucket

```
curl -i --location-trusted -X POST --post301 --data-binary '' -H "Policy-Versioning: enabled"
"{cluster}/mybucket?domain=sample&replicate=immediate"
```

```
HTTP/1.1 201 Created
Location: {cluster}/mybucket?domain=sample
Volume: 32745542cflb9385aaf9b5a701544519
Location: {cluster}/mybucket?domain=sample
Volume: 56e4afcf1d25b43f8672d8f7fe7fbe59
Entity-MD5: U06twgE6BEijmq5+rM+MyA==
Stored-Digest: 534eadc2013a0448a39aae7eaccf8cc8
Last-Modified: Fri, 26 Jun 2015 21:53:27 GMT
Content-UUID: 779a221c6352785eafeef179d70a39d6
Castor-System-Version: 1435355607.358
Etag: "8e66a96c58508b0e256e6beb7c8009d5"
Castor-System-Alias: 779a221c6352785eafeef179d70a39d6
Replica-Count: 2
Date: Fri, 26 Jun 2015 21:53:27 GMT
Server: CASTor Cluster/7.5.1
Content-Length: 46
Content-Type: text/html
Keep-Alive: timeout=14400
```

```
<html>
  <body>New object created</body>
</html>
```

Verify the versioning state of the bucket:

```
curl -i --location-trusted
"{cluster}/mybucket?domain=sample"

HTTP/1.1 200 OK
Castor-System-Alias: 779a221c6352785eafeef179d70a39d6
Castor-System-CID: 157b02492dfb48d86fad8a0935ba440a
Castor-System-Cluster: Sample
Castor-System-Created: Fri, 26 Jun 2015 21:53:27 GMT
Castor-System-Name: mybucket
Castor-System-Version: 1435355607.358
Content-Length: 0
Content-Type: application/x-www-form-urlencoded
Last-Modified: Fri, 26 Jun 2015 21:53:27 GMT
Policy-Versioning: enabled
Etag: "8e66a96c58508b0e256e6beb7c8009d5"
Castor-System-Path: /sample/mybucket
Castor-System-Domain: sample
Volume: 56e4afcf1d25b43f8672d8f7fe7fbe59
Policy-Versioning-Evaluated: enabled
Date: Fri, 26 Jun 2015 21:53:41 GMT
Server: CASTor Cluster/7.5.1
Keep-Alive: timeout=14400
```

Notice `Policy-Version-Evaluated` is also enabled. *All named objects in the bucket are versioned.*

Create a named object in the bucket

These two requests have typical headers.

```
curl -i --location-trusted -X POST --post301 --data-binary '<html><body>First version's content.<
"{cluster}/mybucket/mydir/hello world.html?&domain=sample"
```

```
HTTP/1.1 201 Created
Location: {cluster}/mybucket/mydir/hello%20world.html?domain=sample
Volume: 56e4afcf1d25b43f8672d8f7fe7fbe59
Last-Modified: Fri, 26 Jun 2015 22:02:35 GMT
Entity-MD5: VNvfqzhvwrqWAl3Nc5q0w==
Stored-Digest: 54dbdfab386fc2b82a58097735ce6ad3
Castor-System-Version: 1435356155.156
Etag: "0f1d281546d24412c838058482eae868"
Date: Fri, 26 Jun 2015 22:02:35 GMT
Server: CAStor Cluster/7.5.1
Content-Length: 46
Content-Type: text/html
Keep-Alive: timeout=14400
```

```
<html>
  <body>New object created</body>
</html>
```

Verify the new object:

```
curl -i --location-trusted
"{cluster}/mybucket/mydir/hello%20world.html?domain=sample"
```

```
HTTP/1.1 200 OK
Castor-System-CID: 779a221c6352785eafeef179d70a39d6
Castor-System-Cluster: Sample
Castor-System-Created: Fri, 26 Jun 2015 22:02:35 GMT
Castor-System-Name: mydir/hello%20world.html
Castor-System-Version: 1435356155.156
Content-Length: 39
Content-type: text/html
Last-Modified: Fri, 26 Jun 2015 22:02:35 GMT
Etag: "0f1d281546d24412c838058482eae868"
Castor-System-Path: /sample/mybucket/mydir/hello%20world.html
Castor-System-Domain: sample
Volume: 56e4afcf1d25b43f8672d8f7fe7fbe59
Date: Fri, 26 Jun 2015 22:02:43 GMT
Server: CAStor Cluster/7.5.1
Keep-Alive: timeout=14400
```

```
<html><body>First version's content.</body></html>
```

List versions in the bucket

Elasticsearch must be enabled to use listing operations.



Tip

Always add `&sort=tmborn:DESC,etag:DESC` to listing operations to retrieve the versions in the precise order Swarm is maintaining, starting with the current version.

```
curl -i --location-trusted
"{cluster}/mybucket?format=json&domain=sample&versions&sort=tmborn:DESC,etag:DESC"
```

```
HTTP/1.1 200 OK
Castor-System-Alias: 779a221c6352785eafeef179d70a39d6
Castor-System-CID: 157b02492dfb48d86fad8a0935ba440a
Castor-System-Cluster: Sample
Castor-System-Created: Fri, 26 Jun 2015 21:53:27 GMT
Castor-System-Name: mybucket
Castor-System-Version: 1435355607.358
Policy-Versioning: enabled
X-Timestamp: Fri, 26 Jun 2015 21:53:27 GMT
Last-Modified: Fri, 26 Jun 2015 22:04:40 GMT
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8
Castor-Object-Count: 1
Date: Fri, 26 Jun 2015 22:04:40 GMT
Server: CASTor Cluster/7.5.1
Keep-Alive: timeout=14400
```

```
[
  {
    "last_modified":"2015-06-26T22:02:35.156100Z",
    "hash":"0fld281546d24412c838058482eae868",
    "content_type":"text/html",
    "name":"mydir/hello world.html",
    "bytes":39
  }
]
```

Update the named object

```
curl -i --location-trusted -X PUT --post301 --data-binary '<html><body>Second version's content.<
"{cluster}/mybucket/mydir/hello world.html?domain=sample"
```

```
HTTP/1.1 201 Created
Location: {cluster}/mybucket/mydir/hello%20world.html?domain=sample
Volume: ec8ab948651deb7b9599d2288cf349e6
Last-Modified: Fri, 26 Jun 2015 22:09:26 GMT
Entity-MD5: 0rwew5/zuuYoeLNhLMJkkw==
Stored-Digest: d2bc1ec39ff3bae62878b3612cc24a2b
Castor-System-Version: 1435356566.913
Etag: "d953ada15686af402290cc77780249be"
Date: Fri, 26 Jun 2015 22:09:27 GMT
Server: CASTor Cluster/7.5.1
Content-Length: 54
Content-Type: text/html
Keep-Alive: timeout=14400
```

```
<html>
  <body>New object version created</body>
</html>
```

Verify the change to the named object:

```
curl -i --location-trusted
"{cluster}/mybucket/mydir/hello%20world.html?domain=sample"

HTTP/1.1 200 OK
Castor-System-CID: 779a221c6352785eafeef179d70a39d6
Castor-System-Cluster: Sample
Castor-System-Created: Fri, 26 Jun 2015 22:09:26 GMT
Castor-System-Name: mydir/hello%20world.html
Castor-System-Version: 1435356566.913
Content-Length: 47
Content-type: text/html
Last-Modified: Fri, 26 Jun 2015 22:09:26 GMT
Etag: "d953ada15686af402290cc77780249be"
Castor-System-Path: /sample/mybucket/mydir/hello%20world.html
Castor-System-Domain: sample
Volume: ae4adcf66f67f85ceb8096585fe8aa5e
Date: Fri, 26 Jun 2015 22:13:20 GMT
Server: CASTor Cluster/7.5.1
Keep-Alive: timeout=14400

<html>
  <body>Second version's content.</body>
</html>
```

View the new version in the listing

```
curl -i --location-trusted
"{cluster}/mybucket?format=json&domain=sample&versions"

HTTP/1.1 200 OK
Castor-System-Alias: 779a221c6352785eafeef179d70a39d6
Castor-System-CID: 157b02492dfb48d86fad8a0935ba440a
Castor-System-Cluster: Sample
Castor-System-Created: Fri, 26 Jun 2015 21:53:27 GMT
Castor-System-Name: mybucket
Castor-System-Version: 1435355607.358
Policy-Versioning: enabled
X-Timestamp: Fri, 26 Jun 2015 21:53:27 GMT
Last-Modified: Fri, 26 Jun 2015 22:14:23 GMT
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8
Castor-Object-Count: 2
Date: Fri, 26 Jun 2015 22:14:23 GMT
Server: CASTor Cluster/7.5.1
Keep-Alive: timeout=14400

[
  {
    "last_modified": "2015-06-26T22:09:26.913100Z",
    "hash": "d953ada15686af402290cc77780249be",
    "content_type": "text/html",
    "name": "mydir/hello world.html",
    "bytes": 47
  },
  {
    "last_modified": "2015-06-26T22:02:35.156100Z",
    "hash": "0f1d281546d24412c838058482eae868",
    "content_type": "text/html",
    "name": "mydir/hello world.html",
    "bytes": 39
  }
]
```

INFO the prior version

```
curl -i --location-trusted
"{cluster}/mybucket/mydir/hello%20world.html?domain=sample&version=0f1d281546d24412c838058482eae

HTTP/1.1 200 OK
Castor-System-CID: 779a221c6352785eafeef179d70a39d6
Castor-System-Cluster: Sample
Castor-System-Created: Fri, 26 Jun 2015 22:02:35 GMT
Castor-System-Name: mydir/hello%20world.html
Castor-System-Version: 1435356155.156
Content-Length: 39
Content-type: text/html
Last-Modified: Fri, 26 Jun 2015 22:02:35 GMT
Etag: "0f1d281546d24412c838058482eae868"
Castor-System-Path: /sample/mybucket/mydir/hello%20world.html
Castor-System-Domain: sample
Volume: 56e4afcf1d25b43f8672d8f7fe7fbe59
Date: Fri, 26 Jun 2015 22:05:22 GMT
Server: CASTor Cluster/7.5.1
Keep-Alive: timeout=14400
```

Delete the prior version

Deleting the historical version permanently erases it:

```
curl -i -X DELETE --location-trusted
"{cluster}/mybucket/mydir/hello%20world.html?domain=sample&version=0f1d281546d24412c838058482eae

HTTP/1.1 200 OK
Castor-System-Cluster: Sample
Content-Length: 0
Date: Fri, 26 Jun 2015 22:07:55 GMT
Server: CASTor Cluster/7.5.1
Keep-Alive: timeout=14400
```

List the versions again to verify the deletion:

```
curl -i --location-trusted
"{cluster}/mybucket?format=json&domain=sample&sort=tmborn:DESC,etag:DESC"

HTTP/1.1 200 OK
Castor-System-Alias: 779a221c6352785eafeef179d70a39d6
Castor-System-CID: 157b02492dfb48d86fad8a0935ba440a
Castor-System-Cluster: Sample
Castor-System-Created: Fri, 26 Jun 2015 21:53:27 GMT
Castor-System-Name: mybucket
Castor-System-Version: 1435355607.358
Policy-Versioning: enabled
X-Timestamp: Fri, 26 Jun 2015 21:53:27 GMT
Last-Modified: Fri, 26 Jun 2015 22:24:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8
Castor-Object-Count: 1
Date: Fri, 26 Jun 2015 22:24:52 GMT
Server: CASTor Cluster/7.5.1
Keep-Alive: timeout=14400

[
  {
    "last_modified":"2015-06-26T22:09:26.913100Z",
    "hash":"d953ada15686af402290cc77780249be",
    "content_type":"text/html",
    "name":"mydir/hello world.html",
    "bytes":47
  }
]
```

Delete the current version

Deleting the current version adds a delete marker:

```
curl -i --location-trusted -X DELETE
"{cluster}/mybucket/mydir/hello%20world.html?domain=sample"

HTTP/1.1 200 OK
Castor-System-CID: 779a221c6352785eafeef179d70a39d6
Castor-System-Cluster: Sample
Castor-System-Created: Fri, 26 Jun 2015 22:09:26 GMT
Castor-System-Name: mydir/hello%20world.html
Castor-System-Version: 1435356566.913
Content-type: text/html
Last-Modified: Fri, 26 Jun 2015 22:09:26 GMT
Etag: "d953ada15686af402290cc77780249be"
Content-Length: 0
Castor-System-Path: /sample/mybucket/mydir/hello%20world.html
Castor-System-Domain: sample
Volume: ae4adcf66f67f85ceb8096585fe8aa5e
Date: Fri, 26 Jun 2015 22:27:14 GMT
Server: CASTor Cluster/7.5.1
Keep-Alive: timeout=14400
```

Listing the versions shows the addition of a new version, the delete marker:

```

curl -i --location-trusted
"{cluster}/mybucket?format=json&domain=sample&sort=tmborn:DESC,etag:DESC"

HTTP/1.1 200 OK
Castor-System-Alias: 779a221c6352785eafeef179d70a39d6
Castor-System-CID: 157b02492dfb48d86fad8a0935ba440a
Castor-System-Cluster: Sample
Castor-System-Created: Fri, 26 Jun 2015 21:53:27 GMT
Castor-System-Name: mybucket
Castor-System-Version: 1435355607.358
Policy-Versioning: enabled
X-Timestamp: Fri, 26 Jun 2015 21:53:27 GMT
Last-Modified: Fri, 26 Jun 2015 22:28:16 GMT
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8
Castor-Object-Count: 2
Date: Fri, 26 Jun 2015 22:28:16 GMT
Server: CASTor Cluster/7.5.1
Keep-Alive: timeout=14400

[
  {
    "last_modified":"2015-06-26T22:27:14.643300Z",
    "hash":"86c3f94a9405067f10598b770cb00531",
    "content_type":"text/html",
    "name":"mydir/hello world.html",
    "bytes":0,
    "deletemarker":"true"
  },
  {
    "last_modified":"2015-06-26T22:09:26.913100Z",
    "hash":"d953ada15686af402290cc77780249be",
    "content_type":"text/html",
    "name":"mydir/hello world.html",
    "bytes":47
  }
]
    
```

Notice the delete marker has the same name but is 0 bytes and has "deletemarker":"true".

Check the delete marker

Getting INFO on the current version (the delete marker) results in a **404 Not Found** error, but provides information about the delete marker:

```

curl -I --location-trusted
"{cluster}/mybucket/mydir/hello%20world.html?domain=sample"

HTTP/1.1 404 Not Found
Castor-System-Name: mydir/hello%20world.html
Castor-System-Version: 1435357634.059
Content-Length: 0
Etag: "86c3f94a9405067f10598b770cb00531"
Date: Fri, 26 Jun 2015 22:35:43 GMT
Server: CASTor Cluster/7.5.1
Keep-Alive: timeout=14400
    
```

Note the ETag (version ID) of the delete marker is returned.

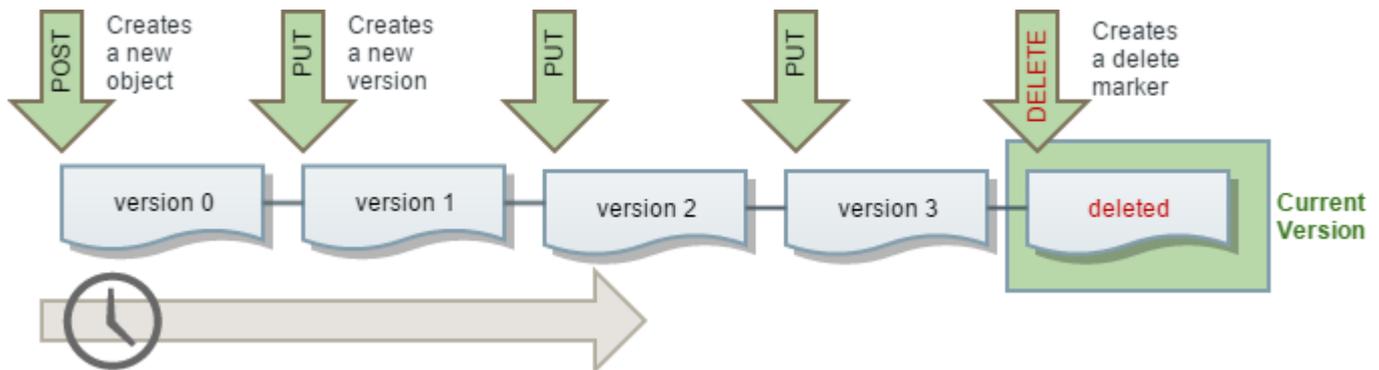
Working with Versioning

- [When Versioning is Enabled](#)
 - [Note](#)
 - [Tip](#)
- [When Versioning is Suspended](#)
 - [Tip](#)

When Versioning is Enabled

When you enable or suspend versioning, the existing objects in your domain or bucket do not change: what changes is how Swarm handles future requests.

Once versioning is enabled in a context, Swarm uses each object's metadata to walk a virtual chain of versions of an object, from creation to deletion:



The order of the versions is the creation order, with the most recent one *always* being the current version.

These virtual version chains are extremely resilient: Swarm linearizes the linkage so the newest version of an object links to the next in the ordering, which links to the next, and Swarm's health processing guarantees temporarily broken chains are repaired. Swarm orders all versions by the time they are written, regardless of the operations that created them.

Note

Only the *sequence* of the version chain is maintained: Swarm does not document change events, such as which operation caused the new version or that one is a restored version of another.

All usual SCSP operations act on the current version, which is the most recently updated one, unless a previous version is mentioned on the request.

Key principles – These principles apply across all versioning operations, whether versioning is enabled or suspended:

- Regular operations always act on the *current* version of the object.
- Every version of each object – *including* a delete marker – is identified by its version identifier, which is its **ETag** value.
- Operations that include the [version query argument](#) always act on the *specific* version alone, even the current version.



Tip

If you get an **HTTP 404 Not Found** or **HTTP 400 Bad Request** error when using the **version** query argument, this indicates that the context is versioning-disabled, which is the default if the versioning policy is still unspecified.

See [Versioning Operations](#) and [Versioning Examples](#).

When Versioning is Suspended

Suspending versioning allows new versions of the same object to stop accruing without jeopardizing the set of versions that already exists.



Tip

If you decide to undo version control altogether, you can disable versioning in the domain or bucket, which triggers the Health Processor to clean up all residual prior versions. This feature is not available in Amazon S3.

When you suspend versioning, existing objects in your domain or bucket do not change: what changes is how Swarm handles future requests. You can re-enable versioning to have Swarm resume versioning behavior where it left off.

- **New Objects** – No versions are created for any objects created after versioning is suspended.
- **Updating Objects** – Swarm retains all existing versions when the versioning state is changed to suspended. After that, any update creates a *new* current version, which is overwritten with updates after that.
- **Retrieving Objects** – Regardless of the versioning state of the domain or bucket, GET Object requests return the current version of an object.
- **Deleting Objects** – If versioning is suspended, a DELETE request creates a delete marker for the object. You can also delete a specified version, which permanently removes that object.

Implementing Versioning

Swarm Versioning is an optional feature introduced in release 8.0; it is modeled on [Amazon S3 Versioning](#) to extend Gateway's support for S3.

- [Guidelines for Versioning](#)
- [Configuring Versioning](#)
- [Managing Policies across Clusters](#)

Guidelines for Versioning

Review these guidelines before implementing versioning in your cluster:

- Plan for higher disk utilization with versioning: each update to a versioned object adds a new object to the cluster (one object updated twice results in three objects stored).
- Make use of [lifepoints](#) to control the lifetime – and thus the cost of storing – multiple versions of your objects.
- For resource management, limit versioning to the specific domains and/or buckets for which it is needed.
- For best performance, enable the [Overlay Index](#) on your cluster.
- When replicating between clusters that *both* enable versioning, avoid unnecessary updates to versioning policies: new policies cannot take effect in the remote cluster until the domain or bucket is replicated, which may take a while.
- Legal hold buckets cannot be versioning-enabled. If using legal hold, the ETag header value (which is the version identifier) needs to be used to hold a previous object version.
- **Upgrades:** Versioning is integrated with Swarm and available on upgrade, but take these steps *before enabling versioning*:
 - Complete the upgrade of both the [storage cluster](#) and the [search cluster](#).
 - If you are using [replication feeds](#), upgrade both the source and target clusters to the same release of Swarm before enabling versioning.

Configuring Versioning

Versioning is one of several content protection features in Swarm, all are controlled by way of *policies*. Swarm evaluates policies for each object based on its cluster and context values:

- Policy-related settings for the **cluster** (*required*)
- Policy-related headers on **domain and bucket** objects (*optional*)

For how to set policies at the cluster level, see [Configuring Cluster Policies](#).

Context

The versioning state of a given object depends on its *context*:

- **Alias object** – context = cluster + domain in which it is tenanted
- **Named object** – context = cluster + domain + bucket



Important

Untenanted alias objects cannot be versioned. Alias objects are *always* tenanted if [cluster.enforceTenancy](#) is enabled in the cluster (which is [required for Gateway](#)).

Policy

Swarm uses the versioning **policy** (set via cluster settings and domain/bucket headers) to determine what versioning-related operations are allowed on each object. The versioning state of the immediate context applies to *every* object in that context, without exception.

Each alias or named object has one of three *versioning states*:

disabled	(default) No versioning exists, so no versions are created; obsolete versions are lost. <i>This state is the normal behavior of Swarm.</i>	If you change the state from enabled to disabled, the health processor erases the prior versions, which are now obsolete.
suspended	No new versions are accumulated but old versions are retained. This is a hybrid between enabled and disabled that preserves history.	If you re-enable versioning from this state, the chain of versions resumes from where it stopped.
enabled	New versions are accumulated as they are created, starting with any version that exists at the time versioning becomes enabled for the object.	

To implement versioning, you need to enable it by setting your versioning policy across all affected container objects, starting with the cluster:

- **Cluster** – set the [configuration parameter](#) `policy.versioning`. (This is a [persisted setting](#), so set it via SNMP.)
- **Domain(s)** – set the domain object's `Policy-Versioning` and `Policy-Versioning-Unnamed` header values.
- **Bucket(s)** – set the bucket object's `Policy-Versioning` header value.

Important

Setting a header does not necessarily enable versioning. For example, you may have added `Policy-Versioning: enabled` to a bucket, but it has no effect if its domain does not have versioning explicitly enabled.

Status settings

The statuses for versioning vary by type of container object:

	Setting	Values	Description
Cluster	Config parameter: <code>policy.versioning</code>	disallowed suspended allowed	Status of versioning within the cluster. Defaults to disallowed. Important: This is a persisted setting , so set it by SNMP, not via configuration file: <pre>snmpset -m +CARINGO-CASTOR-MIB -v2c -M +/usr/share/snmp/mib2c-data -cPASSWORD -Oqs {swarm-ip} clusterConfig.policyVersioning s "allowed"</pre>

Domain	Object header: Policy-Versioning	disabled suspended enabled required	Status of versioning for named objects within the domain, subject to the cluster setting. If the header is missing, the value is considered unspecified and versioning is disabled. Important: To guarantee every bucket in a domain is versioned, set the domain's versioning state to <code>required</code> . Add a <code>Policy-Versioning: {value}</code> header using an SCSP PUT or COPY request on the domain object.
	Object header: Policy-Versioning-Unnamed	disabled suspended enabled	Status of versioning for unnamed objects that are tenanted in the domain, subject to the cluster setting. If the header is missing, the value is considered unspecified and versioning is disabled. Add a <code>Policy-Versioning-Unnamed: {value}</code> header using an SCSP PUT or COPY request on the domain object.
Bucket	Object header: Policy-Versioning	disabled suspended enabled	Status of versioning within the specific bucket, subject to the cluster and domain settings. If the header is missing, the value is considered unspecified and versioning is disabled. Important: If the domain state is <code>enabled</code> , no bucket is versioned until it has its state changed to <code>enabled</code> . Add a <code>Policy-Versioning: {value}</code> header using an SCSP PUT or COPY request bucket object.

You can verify versioning is enabled within a domain or bucket by making a HEAD request and checking the policy-related headers, `Policy-Versioning[-Unnamed]` (which you add) and `Policy-Versioning[-Unnamed]-Evaluated` (which Swarm generates). You control all `Policy-Versioning[-Unnamed]` headers, but Swarm computes and `Policy-Versioning[-Unnamed]-Evaluated` based on the policy of the complete context.

The versioning state of a domain depends on the cluster's state:

	Domain	disabled	suspended	enabled	required (buckets only)
Cluster					
disallowed	disabled	disabled	disabled	disabled	disabled
suspended	disabled	suspended	suspended	suspended	suspended
allowed	disabled	suspended	enabled	enabled	enabled



Note

When the state is unspecified, it defaults to disabled. In a cluster that has versioning allowed, every newly created domain and bucket starts with an unspecified state, so object versioning is disabled until you enable it there explicitly.

The versioning state of a bucket depends on the parent domain's state:

Domain	Bucket	disabled	suspended	enabled
	disabled	disabled	disabled	disabled
suspended	disabled	suspended	suspended	
enabled	disabled	suspended	enabled	
required	enabled	enabled	enabled	

Managing Policies across Clusters

If you are using [replication feeds](#) while using versioning, both the source and target clusters must run the same release of Swarm, but you choose how versioning occurs in the target:

Goal for Target	Target Cluster Policy	Effect
No versioning	(default) <code>policy.versioning = disallowed</code>	In the target cluster, Swarm maintains the current versions (no versioning) and cleans up any obsolete versions that are replicated. The domain and bucket objects replicate with the same policies, but the cluster-wide setting overrides them.
No versioning, access to versions	<code>policy.versioning = suspended</code>	In the target cluster, Swarm does not accumulate versions on local updates but preserves all prior replicated versions.
Identical versioning	<code>policy.versioning = allowed</code>	In the target cluster, Swarm performs versioning identically to the source cluster. The health processors in each cluster do the work of repairing the sequence of prior versions as they expand on both sides. Important: To guarantee faithful replication in your target cluster, change domain and bucket policies <i>before</i> changing the content they contain.
Custom versioning	<code>policy.versioning = allowed</code> and special syntax on custom policies	In the target cluster, Swarm performs versioning identically to the source cluster, except where you configured special context policies. Contact DataCore Support for the specific syntax needed to override policies at the domain or bucket level.

Policies are carried on contexts, and a remote cluster evaluates policies locally based on the contexts it currently holds. Rapid changes to policies in a source cluster with changing content in those domains and buckets returns the correct results, but when those changes are replicated to the target cluster, results may differ.

Versioning Operations

- [Adding Versions](#)
- [Retrieving Versions](#)
- [Listing Versions](#)
- [Deleting Versions](#)
- [Lifepoints for Versions](#)
- [Renaming Versioned Objects](#)
- [Restoring Versions](#)

Note

Operations referencing the current version proceed normally if `version={etag}` is used where versioning is *disabled*, but any other ETag results in an HTTP 404 - Not Found. (v9.2)

Adding Versions

Any write operation (POST, PUT, COPY, APPEND) in a versioning-enabled context creates a new version, and it changes the existing version to be the prior one. (Prior versions are not modified. All write operations create new versions.)

- **POST** stores a brand new object as the current version. (**PUT** can be used if you have `scsp.allowPutCreate` enabled or add the `putcreate` query argument.)
- **PUT** adds a new version of the object, replacing the current version (which remains accessible by ETag); use PUT to update an existing alias. You cannot use PUT with the `version` query argument.
- **COPY ?version={etag}** creates a new version duplicating the *content* of a versioned object with the new headers specified.
- **APPEND ?version={etag}** creates a new version duplicating the *headers* of a versioned object with the new content specified (use 0-length to duplicate the existing content).

Retrieving Versions

Important

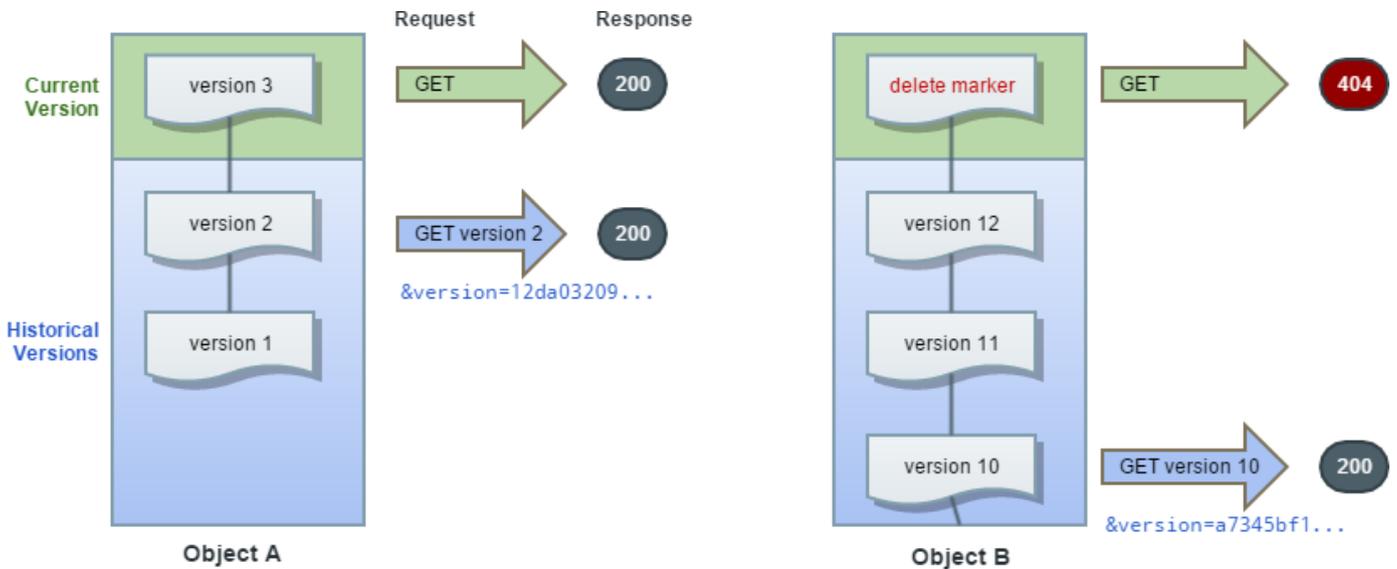
Consider each version being added as being an entire object for the purpose of resource management: it is not a delta from the previous version. Object counts and space utilization reflect 12 objects if you have 12 versions of an object stored (the current and 11 prior).

There are different ways to retrieve current objects versus specific object versions. Add the [version query argument](#) to retrieve a specific version:

- **GET** retrieves the current version of an object.
- **GET ?version={etag}** retrieves a specific version of an object, identified by its ETag header value.
- **HEAD** retrieves the metadata of the current version of an object (and not its content).
- **HEAD ?version={etag}** retrieves the metadata of a specific version of an object, identified by its ETag.

Specify the object's ETag to retrieve the content or metadata of a specific version, which retrieves the specified version of the object. This can be the current version or any prior versions.

An **HTTP 404 Not Found** response including the ETag for the delete marker (not the prior version) is received if a GET or HEAD is issued against delete marker:



Listing Versions

With Elasticsearch indexing your cluster, use it to access listings of your object versions. (Elasticsearch is not required to use versioning.)

Versioning headers – Swarm returns these additional headers with historical (non-current) versioned objects:

- **uuidNextVersion** – Points to the primary UUID of the object that is the newer version. This field transfers during replication within the cluster but not to remote clusters (each cluster resolves its own chains).
- **tmNextVersion** – Holds the birth date of the newer (uuidNextVersion) object (they update as a pair). This field transfers during replication within the cluster but not to remote clusters.
- **tmDeleted** – Holds the death date of this particular version. This field *does* transfer to remote clusters so the needed feed processing and clean up occurs.



Differences from S3

- Unlike Amazon S3 (which retrieves only a maximum of 1000 objects), Swarm has no such limits, nor does it break batches on version boundaries.
- With rapid updates of versioned objects, there is a delay in listing consistency, which can cause 409 Conflict and 503 Service Unavailable responses. Using `replicate=immediate` does not prevent 409 responses; a 1-second delay always applies to versioned buckets.

Add the **versions** query argument to the GET Bucket request to list all versions of *all objects* in a bucket.

Use **prefix** query argument along with **versions** query argument to the GET Bucket request to list all versions of *one* object, which limits the set of versions returned to those related to the object.

```
GET /mybucket?domain=sample&format=json&versions HTTP/1.1
GET /mybucket?domain=sample&format=json&versions&prefix=objectName HTTP/1.1
```



Tip

Add `&sort=tmborn:DESC,etag:DESC` to the listing operations to verify retrieval of the versions in the precise order Swarm is maintaining, starting with the current version. The secondary sort, ETag, is Swarm's tie-breaker for rapid updates in which two versions have the same tmborn.

Deleting Versions

The behavior is the same when versioning is enabled or suspended:

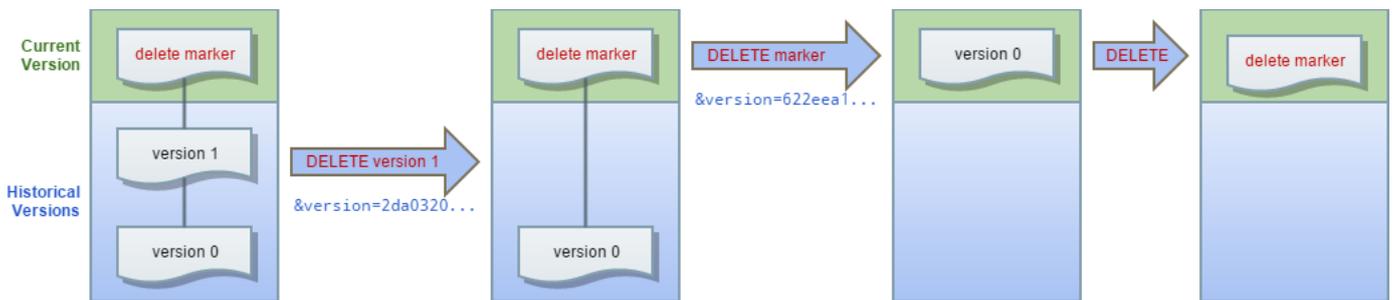
- **Deleting without specifying a version:** Swarm operates on the current version and creates a delete maker.
- **Deleting with a version specified:** Swarm permanently pushes the version out of existence, even if it is the current version.

Important

A specific version of an object is removed permanently from the sequence of versions *and* from the cluster when deleted. These objects cannot be recovered.

A DELETE does not permanently delete an object when versioning is enabled. Swarm inserts a delete marker instead, and the marker becomes the current version of the object with a new ID.

The response to a DELETE operation includes the ETag of the version being deleted, not that of the delete marker:



Use DELETE Object and specify the ETag to permanently delete versioned objects:

```
DELETE /mybucket/photo.gif?version=c347edc55b3c4fadb76d93022a29b07a HTTP/1.1
```

Handling Delete Markers

A delete marker is a placeholder (marker) for a versioned object named in a DELETE request. Because the object was in a versioning-enabled bucket, the object was not deleted. The delete marker causes Swarm to operate as if it had been deleted. A delete marker has a name and version ETag like any other object, but it differs from other objects in several ways:

- It has no data.
- Its only operation is DELETE, which only the owner can request.
- It has nominal size.

Swarm can create a delete marker, and it does so when a DELETE Object request is sent on a versioning-enabled/suspended object . The object named in the DELETE request is not actually deleted. Instead, the delete marker becomes the current version of the object. (The object's name becomes the name of the delete marker.)

The version ETag in a DELETE Object request must be included to permanently delete a delete marker. Owners can permanently delete these markers while the context is versioning-enabled/suspended. (The health processor cleans up the markers along with the prior versions if the context returns to versioning-disabled.)

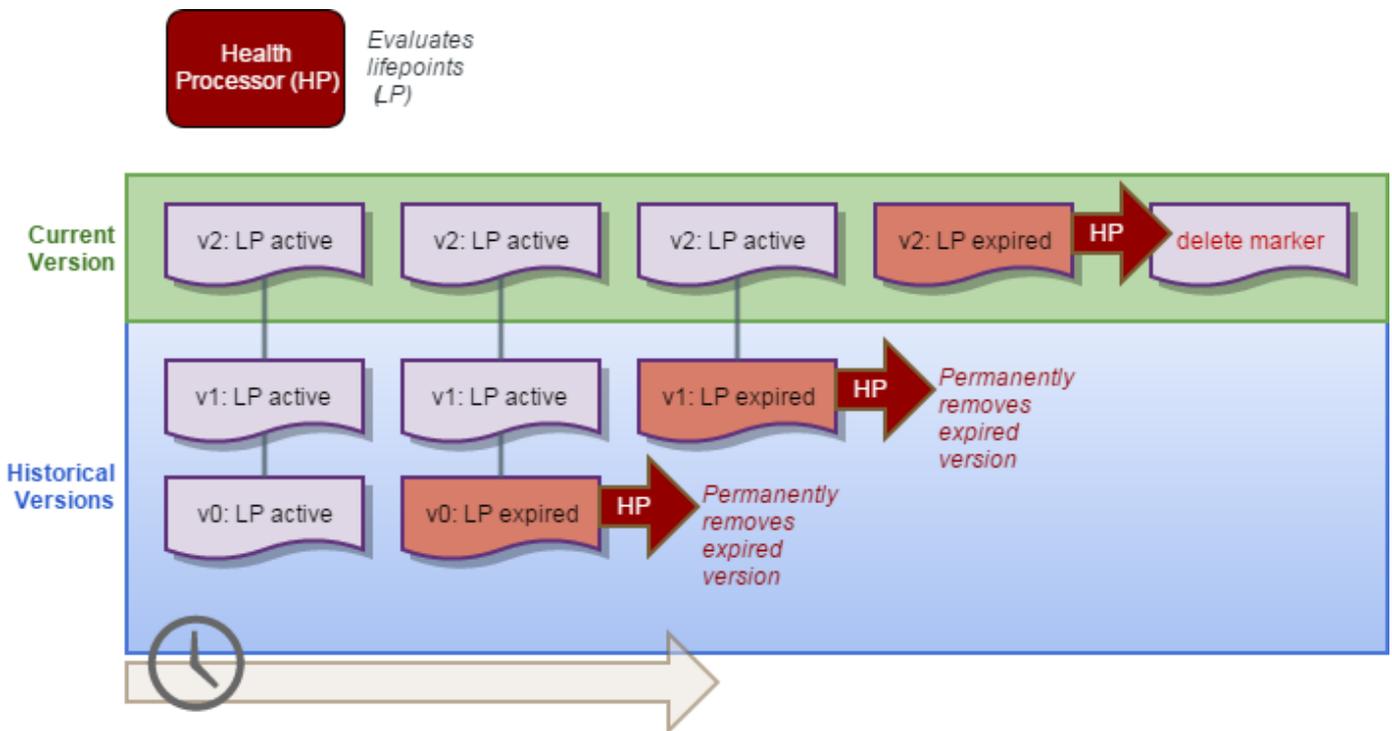
Note

The effect of removing the current delete marker is the prior version of the object becomes the current version.

Lifepoints for Versions

Object versions can be deleted on demand, but lifepoints can be defined for objects with a well-defined lifecycle to have Swarm permanently remove prior object versions as they expire. The Health Processor cleans up (erases) versions with expired lifepoints for both versioning-enabled and versioning-suspended states, deleting particular object versions. The Health Processor repairs the gap in the version sequence caused by the deletion.

The Health Processor creates a delete marker that becomes the current version when the lifepoint expires on the *current* version of the object: it does not resurrect a previous version.



The Health Processor acts on versioned objects in these ways:

- Evaluates lifepoints of current versions, performing EC-related conversions or lifepoint deletions.
- Keeps the desired number of replicas of versioned objects.
- Maintains the current version of an object and the chain of version linkages.
- Deletes prior versions of objects no longer accessible.

Note

Note

The Health Processor does not perform any erasure coding (EC) conversions, even on current versions, if the object is `versioning-enabled` or `versioning-suspended`.

Renaming Versioned Objects

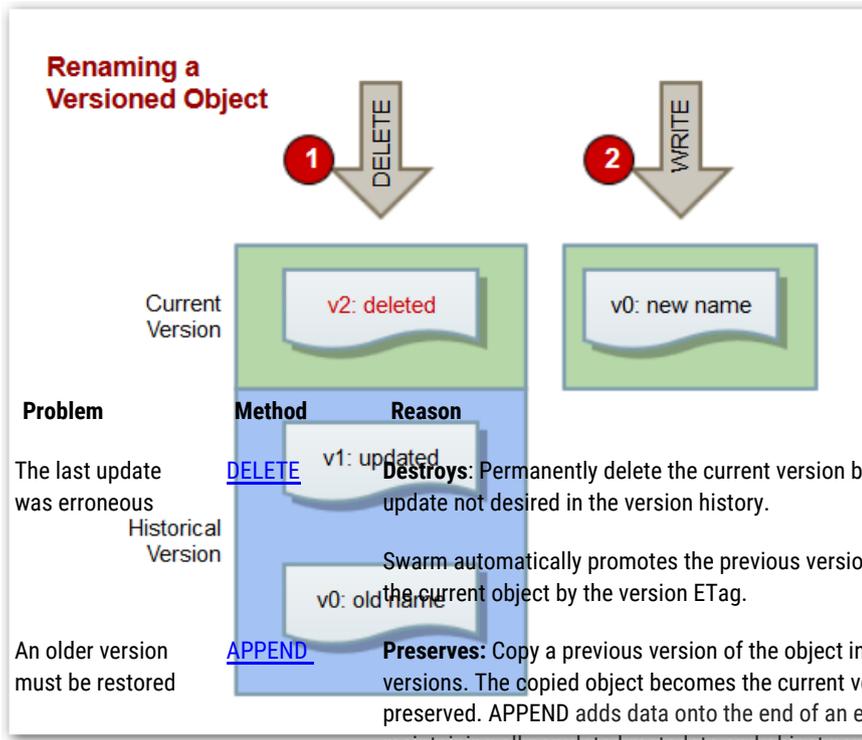
Renaming a versioned object is performed by ending the old one and starting a new one. There are *two requests and two objects*:

1. a DELETE on the old name, which terminates the version chain with a delete marker
2. a WRITE of a new object, with the new name

Note: the name change can be reversed by deleting the new object, reading the last good version, and appending the version.

Restoring Versions

A key benefit of versioning is the ability to restore previous versions of an object. There are several ways to restore content by referencing specific versions, depending on your situation:

Renaming a Versioned Object		
		
Problem	Method	Reason
The last update was erroneous	DELETE	Destroys: Permanently delete the current version by ETag: <code>version={etag}</code> to undo an erroneous update not desired in the version history.
Historical Version		Swarm automatically promotes the previous version to be the current version of the object when deleting the current object by the version ETag.
An older version must be restored	APPEND	Preserves: Copy a previous version of the object into the same context to restore any one of the prior versions. The copied object becomes the current version of the object, and all object versions are preserved. APPEND adds data onto the end of an existing named object or aliased object while maintaining all populated metadata and object name or alias UUID.
		Use a 0-length APPEND to duplicate the prior version <i>exactly</i> without change to content or metadata.
		Any earlier version can be made the current version by copying a specific version of the object into the same bucket because all object versions are preserved. Swarm supplies a new ID and it becomes the current version of the object. A subsequent GET retrieves the copy.
New metadata needs to be added	COPY	Replaces: Use COPY to duplicate the prior version with updates to the metadata. COPY updates the metadata on an existing object by copying the content verbatim while replacing the metadata.
		This technique facilitates extending the custom object metadata over time, to improve the scope and usefulness of your Elasticsearch queries.

An older version must be updated	PUT	<p>Invalid: Prior versions are historical and cannot be altered (only deleted), so PUT cannot be used with them. Use COPY or APPEND on the older version, including the changes needed.</p> <p>Use DELETE to remove the specific version permanently if the existence of the incorrect version is a concern or a liability.</p>
----------------------------------	----------------	--

Erasure Coding EC

Replication is a proven and valuable mechanism to verify data integrity, but the cost per GB of storage can become high as object sizes and cluster sizes expand. A complementary data protection strategy, erasure coding (EC), provides high data durability with a smaller footprint. Swarm manages EC and replication together to optimize cost-effectiveness, converting objects between them seamlessly and dynamically, based on the policies set.

- [How EC works](#)
- [How much EC protects](#)
- [How much disk space EC saves](#)
 - [Note](#)

How EC works

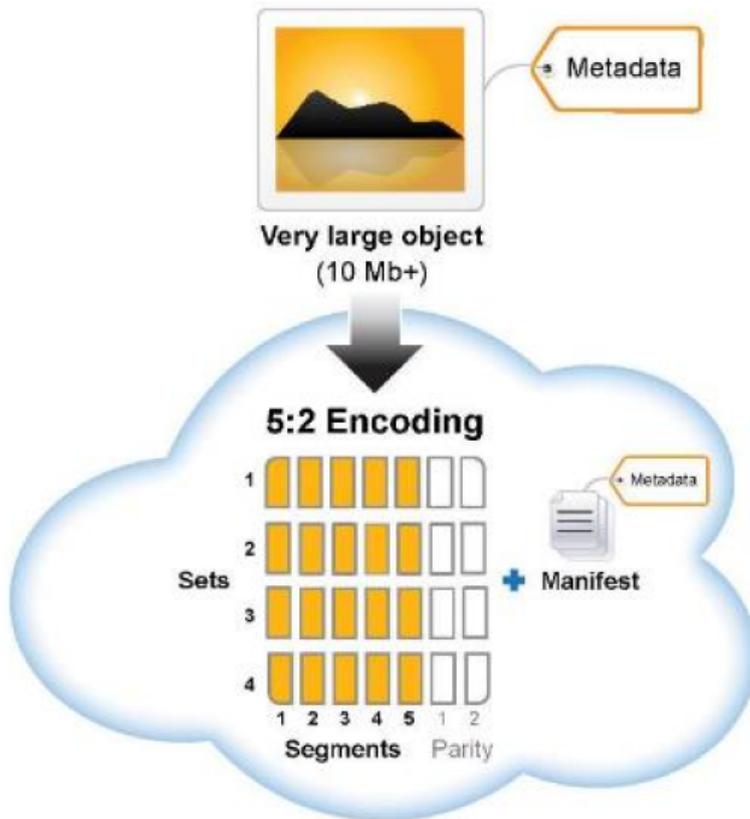
Erasure coding breaks the original object into multiple **data segments (k)** and computes additional **parity segments (p)** based on the content of the data segments. This results in m total segments (**$k + p = m$**) being distributed to m different nodes (or subclusters) in the storage cluster (see the [ec.protectionLevel](#) setting).

The erasure coding encoding level is expressed as a tuple in this format:

```
{data segments}:{parity segments}
```

Swarm creates multiple sets of erasure segments for *very* large objects. The object breakdown into one or more erasure sets is transparent to external applications. A GET or HEAD of an erasure-coded object uses the same syntax as a replicated object.

The following illustration represents how erasure coding works:



How much EC protects

Swarm still reads the object as long as there are still k total segments (any combination of original data or parity) remaining in the cluster if a hard disk or a node containing an erasure segment fails. Protection against disk failure for the object is equal to the number of specified parity p segments.

The segments from a 5:2 (5 data segments with 2 parity segments for a total of 7 segments) or 8:2 (8 data and 2 parity segments for 10 total segments) erasure code are protected against the loss of any two nodes because they are distributed to different nodes. An erasure-coded object is immediately retrievable when accessed even if some segments are missing. Regenerating the missing erasure set segments is performed in a self-healing, cluster-initiated manner (similar to the recovery process for replicated objects) to protect against further disk loss. This process kicks off automatically when a missing volume is detected and automatically regenerates any missing segments.

Swarm always seeks the widest possible distribution of content (whether replicas or EC segments) *regardless of settings*, to maximize data protection. Swarm applies the following $2p+2$ protection levels as possible:

1. One segment per subcluster ... up to p segments per subcluster
2. One segment per node ... up to p segments per node
3. One segment per volume

How much disk space EC saves

The amount of disk space (or *footprint*) used for erasure-coded objects depends on the ratio of data to parity segments in the specified encoding.

Use the following formula to roughly calculate the disk space expected to be used by an EC object with one set of erasure segments:

(total segments ÷ data segments) × object size	= object footprint
$((k+p) \div k) \times \text{GB}$	= total GB

How footprint changes with different EC encoding (versus 3 reps)

- 1 GB object with 5:2 encoding: $((5 + 2) \div 5) \times 1 \text{ GB} = 1.4 \text{ GB}$ (vs. 3 GB for replication)
- 3 GB object with 5:2 encoding: $((5 + 2) \div 5) \times 3 \text{ GB} = 4.2 \text{ GB}$ (vs. 9 GB for replication)
- 3 GB object with 7:3 encoding: $((7 + 3) \div 7) \times 3 \text{ GB} = 4.3 \text{ GB}$ (vs. 9 GB for replication)



Note

Additional system metadata is written with each EC segment which adds about 16 bytes per segment.

See [Hardware Requirements for Storage](#) for how to size and optimize hardware for erasure coding.

See [Cluster Protection Planning](#) for how to determine the required number of nodes and optimize EC performance in a cluster.

- [Troubleshooting Erasure Coding](#)
- [Implementing EC Encoding Policy](#)
- [Methods Affected by Erasure Coding](#)
- [Conversion between Content Protection Types](#)

Troubleshooting Erasure Coding

- [Tip](#)
- [Note](#)
- [Bad Request](#)
- [Replicated, Not Erasure-Coded](#)

Tip

When troubleshooting encoding and policy issues, add the [verbose query argument](#), which returns all relevant headers in the response.

The following diagram details how Swarm evaluates protection requests:

Note

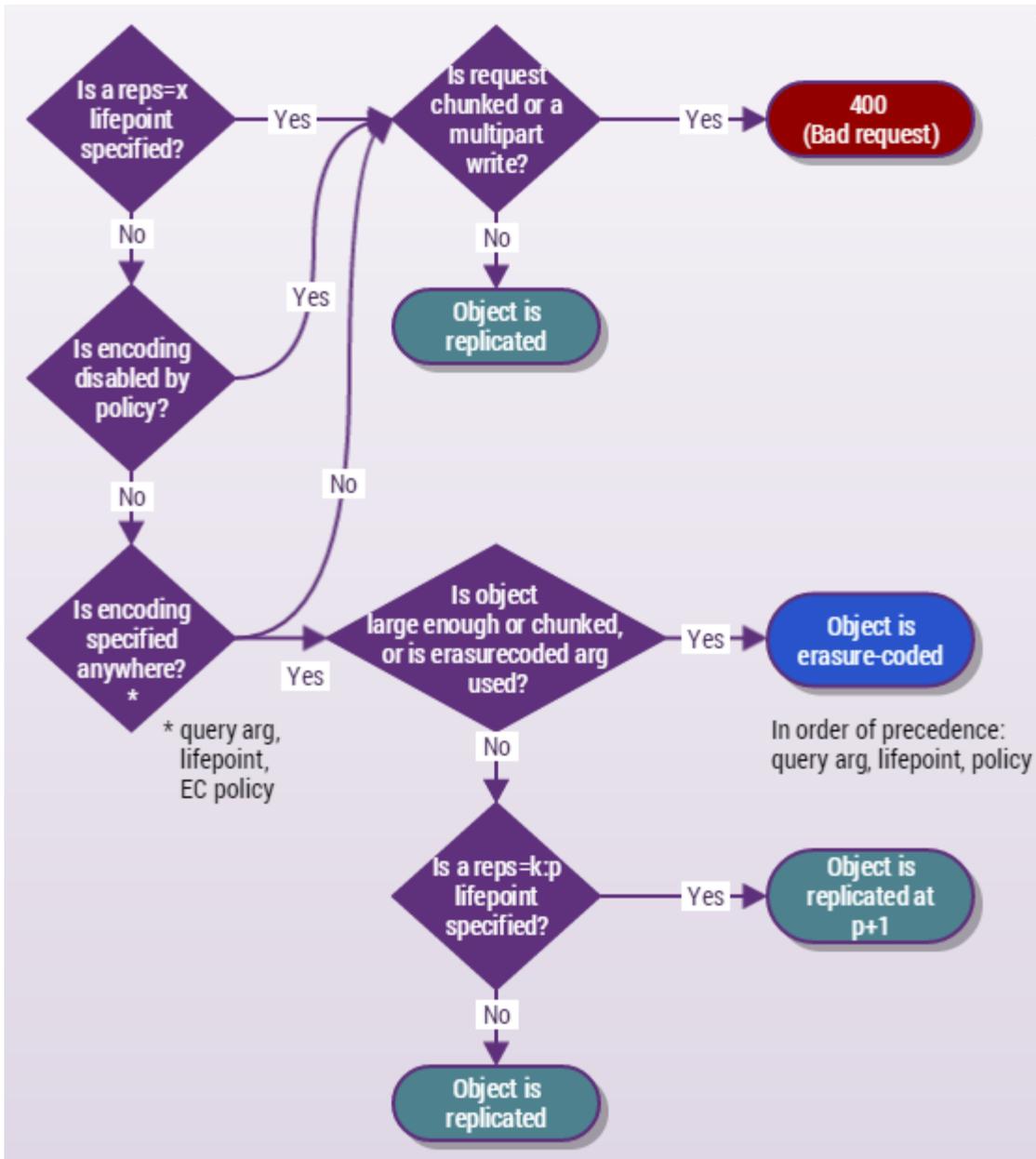
Note

Every multipart write must be erasure coded for upload; if the uploaded object does not meet the current policy for EC encoding, the Health Processor converts it to a replicated object. To maintain erasure coding for the lifetime of the object, add a lifepoint to that effect.

Bad Request

Both chunked requests and [Multipart Write](#) operations must be erasure-coded to succeed. An HTTP 400 (Bad Request) results from such a request having one of these problems:

- The request includes a `reps=x` lifepoint (which specifies number of replicas, not encoding).



- Encoding is disabled, which overrides any query argument or lifepoint encoding.
- Encoding is not specified anywhere.

Replicated, Not Erasure-Coded

If encoding is allowed, and specified in the lifepoint, but the object is too small to qualify, it is replicated at $p+1$.

For example, if 5:2 encoding was specified, 3 replicas are written.

Implementing EC Encoding Policy

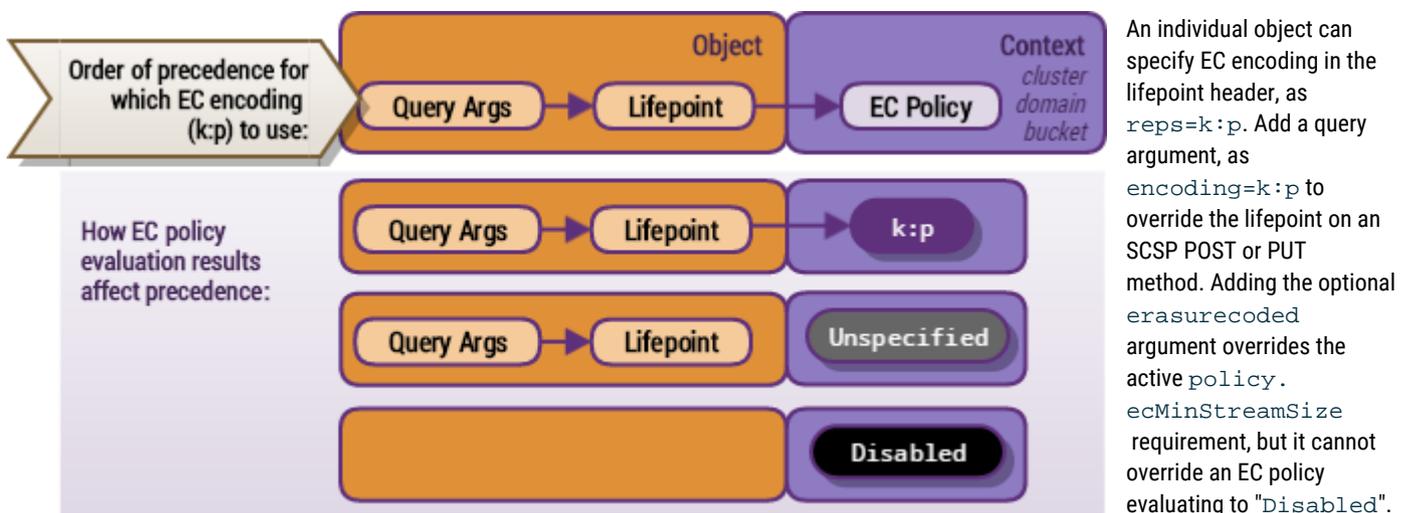
- [Options for specifying EC encoding](#)
 - [Tip](#)
 - [Disable erasure-coding cluster-wide](#)
 - [Require objects to specify encoding, regardless of context:](#)
- [How Swarm resolves EC policies](#)
- [Example EC policy resolution](#)
 - [Tip](#)
- [How to view and set EC encoding policy](#)
 - [Important](#)
 - [Exception](#)
 - [Writing a domain policy](#)
 - [Writing a bucket policy](#)

Context-level policies can be created to manage EC (erasure coding) across the cluster. Swarm ships with cluster-level EC encoding unspecified. Unless EC is specified at the object level, every object is fully replicated, regardless of size.

Set custom EC policies for the encoding ($k:p$) and the object size minimums applicable to specific domains and buckets (contexts). Context-level policies allow building a wide range of erasure-coding controls to meet business needs:

- Disable EC on a bucket, forcing it to perform full replication on every object
- Quadruple the object size triggers erasure coding in a special-use bucket
- Set different EC encodings for named versus unnamed objects in a specific domain
- Lock the domain policy as "unspecified" to cancel any policies bucket owners may have set, forcing them to accept the current cluster policy
- Lock the cluster policy as "disabled" to stop *all* erasure coding, voiding any query argument or lifepoint encodings passed in with the request

It is necessary to understand how Swarm chooses among different encoding ($k:p$) to design EC policies because Swarm allows specifying them at multiple levels simultaneously. Encoding specified at the object level wins over any held by the context; the context policy *evaluation* can disable the encoding requested at the object level:



See [Lifepoint Metadata Headers](#) for setting lifepoints with erasure coding.

See [SCSP Query Arguments](#) for using erasure coding query arguments.

Options for specifying EC encoding

A different EC encoding policy can be defined at the cluster and each context (domain and bucket) level, using this 2-part setting:

Part	Scope	Value (string)	Notes	Example
Encoding	Affects the <i>current</i> context level.	unspecified (default)	Allows erasure coding at this level and below, using the available policy or cluster defaults.	<code>unspecified</code>
	Returns 400 Bad request if value is missing or out of range	disabled	Disables erasure coding at current level.	<code>disabled</code>
		k:p <i>{data segments}</i> : <i>{parity segments}</i>	The specific EC encoding to use at this level. Requirements: <ul style="list-style-type: none"> • $1 \leq k$ • ec.minParity $\leq p$ • $k + p \leq \text{policy.replicas max}$ Subject to the resolved policy's <code>eCMinStreamSize</code> value unless it is overridden by the erasurecoded argument .	<code>7:3</code>
Lock (optional)	Affects any contexts <i>below</i> the current context.	anchored	Applies this encoding to all levels below, overriding any lower-level policies.	<code>disabled anchored</code> <code>7:3 anchored</code>

EC encoding is subject to policy resolution on each write request, evaluating the object-level encoding with the context. For contexts, Swarm uses the policy at the given level unless a level above it is anchored. Swarm uses the anchored encoding.



Tip

The "anchored" lock exists to allow suppressing any policies added by domain or bucket owners below.

Anchoring allows enforcing these two situations:

Disable erasure-coding cluster-wide

```
policy.ecEncoding = disabled anchored
```

Require objects to specify encoding, regardless of context:

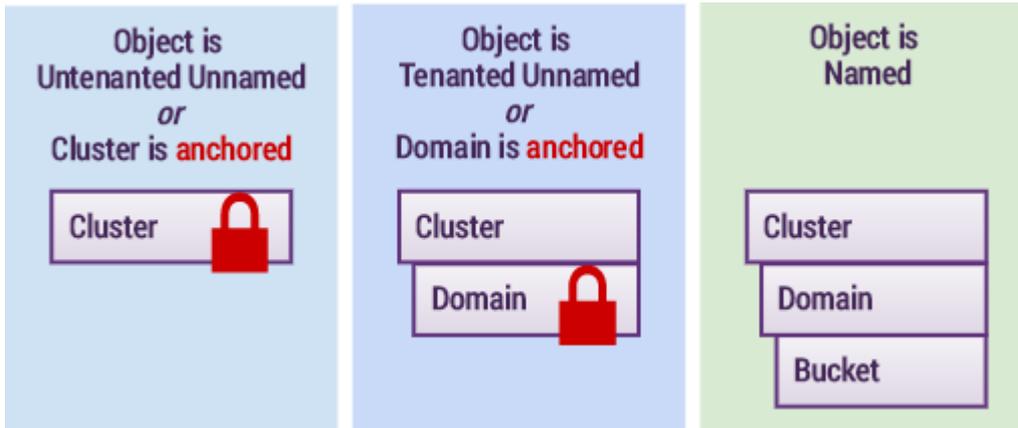
```
policy.ecEncoding = unspecified anchored
```

How Swarm resolves EC policies

Swarm begins EC policy evaluation by eliminating contexts (domain and bucket) not applicable, for one of two reasons:

- because of the type of object

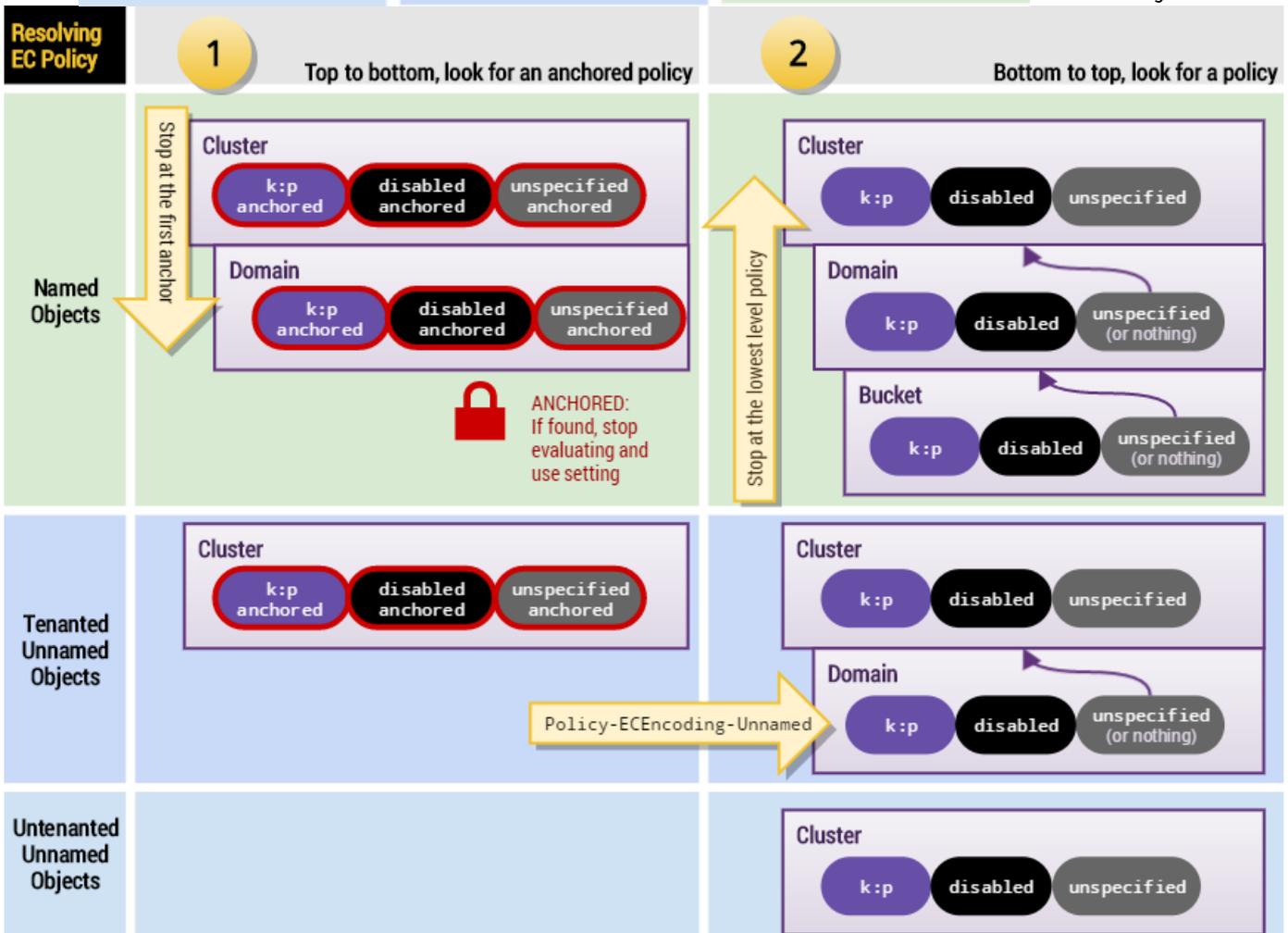
- because of the "anchored" override



Swarm (1) selects the anchored policy if it exists, or else (2) checks the relevant contexts from the bottom up and selects the first policy it finds:

Example EC policy resolution

For example, consider a set of conflicting context-level



policies such as this:

cluster	Configuration setting	policy.ecEncoding	7:3
----------------	-----------------------	-------------------	-----

	domain	Persisted header	Policy-ECEncoding	5:2 anchored
			Policy-ECEncoding-Unnamed	5:2
	bucket	Persisted header	Policy-ECEncoding	disabled

The EC policy evaluation resolves differently depending on the type of object being written:

Object Type	Evaluates to	Explanation
named	5:2	The bucket setting is ignored because the domain setting is anchored.
tenanted unnamed	5:2	The bucket setting is ignored because unnamed objects cannot reside in a bucket. The Policy-ECEncoding-Unnamed needs to be defaulted to the cluster value if it is unspecified.
untenanted unnamed	7:3	The domain setting is ignored because untenanted objects cannot reside in a domain.



Tip

See the diagram in [Troubleshooting Erasure Coding](#) to see how Swarm resolves the protection policy (whether to use erasure coding or replication) on a request.

How to view and set EC encoding policy

EC Encoding is one of several content protection features in Swarm, all are controlled by way of *policies*. Swarm evaluates policies for each object based on cluster and context values:

- Policy-related settings for the **cluster** (*required*)
- Policy-related headers on **domain and bucket** objects (*optional*)

For setting policies at the cluster level, see [Configuring Cluster Policies](#).

Creating a policy involves changing *one or both* EC requirements for the given scope:

ecEncoding – Whether to allow/prevent/change the EC encoding

ecMinStreamSize* – Whether to change the object size triggering erasure-coding.



Important

For efficiency, ecMinStreamSize* is given in units of *nMB* and *nGB*, *not bytes*, as are the old settings. Convert (not copy) the old ec.minStreamSize value to the new setting specifying units. Note: 1 MB = 1,000,000 bytes.



Exception

The `ecMinStreamSize` limit does not apply if the request is to initiate a parallel upload (which *must* be erasure-coded).

Set encoding policies with the appropriate naming and method, starting with the cluster, once decided on the encoding policy for each level:

Scope	Type	Persisted Name, SNMP MIB	Configuration Method
cluster	Configuration setting	<p>policy.ecEncoding, policyECEncoding</p> <p>policy.ecMinStreamSize, policyECMinStreamSize</p>	<p>Set the cluster parameters specific to EC encoding policy and verify the other EC-related settings.</p> <p>Use the <code>snmpset</code> command on the cluster's settings object to change and persist the cluster-wide policies. See Using SNMP with Swarm.</p> <p>See Configuring Cluster Policies.</p>
domain	Persisted header	<p>Policy-ECEncoding, policyECEncoding</p> <p>Policy-ECMinStreamSize, policyECMinStreamSize</p> <p>Policy-ECEncoding-Unnamed, policyECEncodingUnnamed</p> <p>Policy-ECMinStreamSize-Unnamed, policyECMinStreamSizeUnnamed</p>	<p>Use a write method (POST for new, COPY for update) on the domain object to set policy values.</p> <p>Use a read method (GET, HEAD) on the domain object to check policy values.</p> <p>Separate encoding can be specified for unnamed objects, as if they are in a separate bucket, because they are tenanted directly in domains:</p> <ul style="list-style-type: none"> • Set <code>Policy-ECEncoding-Unnamed</code> to allow/prevent/change erasure-coding of <i>unnamed</i> objects in this domain. • Set <code>Policy-ECMinStreamSize-Unnamed</code> to change the size to trigger erasure-coding of <i>unnamed</i> objects in this domain.
bucket	Persisted header	<p>Policy-ECEncoding, policyECEncoding</p> <p>Policy-ECMinStreamSize, policyECMinStreamSize</p>	<p>Use a write method (POST for new, COPY for update) on the bucket object to set policy values.</p> <p>Use a read method (GET, HEAD) on the bucket object to check policy values.</p>

This new domain has a separate required policy for named objects (which bucket owners are not able to override), but unnamed objects follow the current cluster-level policy:

Writing a domain policy

```
curl -iL -XPOST --post301 --data-binary ""  
-H "Policy-ECMinStreamSize: 2MB"  
-H "Policy-ECEncoding: 7:3 anchored"  
-H "Policy-ECEncoding-Unnamed: unspecified"  
"http://{cluster}?domain=myDomain"
```

This new bucket disables erasure-coding, but this policy is canceled by any policies "anchored" above it at the domain or cluster level:

Writing a bucket policy

```
curl -iL -XPOST --post301 --data-binary ""  
-H "Policy-ECEncoding: disabled"  
"http://{cluster}/myBucket?domain=myDomain"
```

Methods Affected by Erasure Coding

Following is how the SCSP methods are affected by erasure-coded objects:

Method	Description
<u>POST</u>	A new object written to the storage cluster is erasure-coded if it meets the <u>EC criteria</u> .
<u>PUT</u>	As with POST, if a non-erasure-coded object is PUT and the data causes the object to meet the <u>EC criteria</u> , the object can be erasure-coded.
<u>APPEND</u>	<p>As with PUT, if data appended to a non-EC object causes the object to meet the <u>EC criteria</u>, the object can be erasure-coded.</p> <p>APPENDs are optimized to POST a new set of EC segments with the appended data and PUT the manifest for a previously erasure-coded object, instead of rewriting the whole object to include the appended data as with replicated objects. Data appended to an object not previously erasure-coded can cause the object to become erasure-coded if the additional appended data pushes the object size over the configured <code>ECMinStreamSize</code> threshold.</p> <p>Attempting to use the argument <code>erasurecoded</code> without <code>encoding</code> or when the cluster is not configured for EC results in an HTTP 400 Bad Request error.</p> <p>On an APPEND for an existing EC object, the new segments are encoded with the parameters of the first existing erasure set.</p>
<u>COPY</u>	Instead of rewriting the entire object to PUT the metadata, COPY for erasure-coded objects PUTs the metadata on the manifest only without rewriting any content data.
<u>HEAD</u>	Returns the Manifest: ec header for an erasure-coded object.
<u>GET</u>	The cluster automatically retrieves all segments, so a GET does not operate differently for an erasure-coded object.
<u>DELETE</u>	Swarm deletes both the manifests and the segments for the object.

Conversion between Content Protection Types

- [Lifepoint conversions](#)
- [Encoding conversions](#)
- [Using cache coherency headers after erasure coding](#)

Lifepoint conversions

The Health Processor maintenance mechanism converts objects as necessary between:

- Standard replication and erasure coding
- Erasure coding and standard replication

The Health Processor replicates the logical object back into the cluster with the new replication scheme when you configure your cluster settings for this type of conversion. The object is visible with some header changes, and the new object version supersedes the older version after the conversion. The source cluster is re-replicated to the target cluster as a new version of the object if the conversion is performed in a source cluster.

Content protection can change due to a set of explicitly specified lifepoint policies over time. An explicit lifepoint specification that includes a **reps=** value—either as a whole number or a colon-separated k:p EC encoding—takes precedence over any default cluster setting. This *explicit conversion* responds to explicit lifepoint specifications. Explicit conversions include replication to erasure coding, erasure coding to replication, and erasure coding to another erasure-encoding scheme.

Explicit conversions occur on the next Health Processor cycle following the lifepoint change.

Encoding conversions

In addition to lifepoint conversions, the Health Processor performs encoding conversions when:

- The cluster includes a **policy.ecEncoding** value, which sets the number of data and parity segments to be used when erasure coding objects (for example, 5:2).
- The object is greater than the **policy.ecMinStreamSize** value, which indicates the minimum size (in bytes) for an object to be automatically erasure-coded.

If the object is replicated wholly, **policy.ecEncoding** is specified, and the object size is greater than the **policy.ecMinStreamSize** value, the object is converted to erasure coding. This *implicit conversion* occurs because of your cluster settings. Implicit conversions are used to convert legacy data—perhaps without lifepoints—to the default cluster encoding scheme, enabling legacy data to take advantage of the new capability. An object remains replicated at **scsp.defreps** replicas if the object is replicated and **policy.ecEncoding** is not configured or the object size is less than the **policy.ecMinStreamSize** value.

The Health Processor converts these objects at a slower rate than the next Health Processor cycle to balance its processing cycles between object conversions and other system requirements. The **ec.conversionPercentage** setting governs the conversion rate. The **ec.conversionPercentage** setting defaults to zero, which implies no implicit conversion. Until the object is converted, it is replicated with **scsp.defreps** reps.

Using cache coherency headers after erasure coding

When the Health Processor converts objects from standard replication to erasure coding, it replaces the **Etag** and the **Castor-System-Version** header. When the **if-match** and **if-none-match** cache coherency headers compare the request header content against the **Castor-System-Created** header on the object (which does not change during the conversion), the headers believe the object has changed, even though the content data is actually the same. The cache coherency headers do not always operate as expected.

To achieve consistent results,

- Use **if-modified-since** and **if-unmodified-since** headers after erasure-coding conversion or during remote replication where either the original or destination object is erasure-coded.
- Do not use *only* the **Etag** header after an erasure-coding conversion because the header may inadvertently change during an object conversion.

Replication

- [Note](#)
- [How Replication Affects Risk](#)
- [Controlling Replication Protection](#)
 - [Caution](#)
 - [Deprecated](#)
- [Increasing Replication Priority](#)
- [Enforcing Replicate On Write \(ROW\)](#)

Swarm can provide protection on disk by creating multiple copies of each object on different nodes called *replicas*. Control how many replicas are created for each object and how quickly they are created after the object is initially stored in the cluster.

 **Note**

There is one instance of an object in the cluster if one object replica exists in a cluster. In this context, *replica*, *instance*, and *object* are all synonymous.

How Replication Affects Risk

By default, each object in Swarm is stored with two replicas, with each replica residing on a different node in the cluster. Replicas are distributed across subclusters if a cluster is configured to use subclusters.

In the event of a total failure or a hard drive fails for any reason, the cluster reacts quickly and initiates a volume recovery process for each missing drive. The recovery process rapidly creates additional replicas elsewhere in the cluster of all objects stored on the now missing drive(s) so each object again has two replicas. There is a protection risk for the sole replica of the object in the cluster if a second drive fails before the recovery process is completed.

There can also be a potential period of vulnerability at the moment an object is first stored on Swarm if Replicate On Write option to create multiple *simultaneous* replicas is not used.

Controlling Replication Protection

A rapid sequence of drives to fail is possible, but unlikely. The solution is to change the replication requirements if this presents an unacceptable risk for an application. Changing the default replication requirements to a greater number of replicas allows a trade of disk space savings for added security.

 **Caution**

Specifying too many replicas relative to nodes has consequences. Setting the number of replicas equal to the number of storage nodes can lead to uneven loading when responding to volume recoveries.

To set the replication protection for the cluster, configure a single setting, `policy.replicas`, with three required parameters, for `min`, `max`, and `default` number of replicas:

```
policy.replicas: min=2 max=5 default=3
```



Deprecated

The [cluster setting policy.replicas](#) replaces the following three, which are all [deprecated](#): `scsp.minReplicas` `scsp.maxReplicas` `scsp.defaultReplicas`

See [Implementing Replication Policy](#).

Increasing Replication Priority

By default, Swarm writes a new object to one node, responds to the application with a success code and UUID (or name), and then quickly replicates the object as needed to other nodes or subclusters. The replication step is performed as a lower priority task.

While this creates the best balance of throughput and fault tolerance in most circumstances, there are cases where you may want to provide the replication task the same priority as reads and writes, which guarantees replication occurs quickly even under heavy sustained loads.

The cluster administrator can add the following parameter to the node or cluster configuration file:

```
health.replicationPriority = 1
```

With replication set to priority 1, object replication is interleaved in parallel with other operations. This may have a negative impact on cluster throughput for use cases involving sustained, heavy writes. With `health.replicationPriority = 1`, it is still possible (though much less likely) that the failure of a node or volume can cause some recently written objects to be lost if the failure occurs immediately after a write operation but before replication to another node can be completed.

Enforcing Replicate On Write (ROW)

Another replication strategy to protect content is *Replicate on Write* (ROW).

Without ROW, the client writes a single copy and depends on the Health Processor (HP) to create the necessary replicas. Relying on HP leaves open a small window for data loss: the volume containing the node holding the sole copy can fail before HP completes replication. ROW eliminates the window by guaranteeing all replicas are written on the initial request.

How it works: The ROW feature requires Swarm to create replicas in parallel before it returns a success response to the client. ROW protection applies to `WRITE`, `UPDATE`, `COPY`, and `APPEND` requests. The secondary access node (SAN) sets up connections to the number of available peers required to create the needed replicas when ROW is enabled.

See [Configuring Replicate On Write](#).

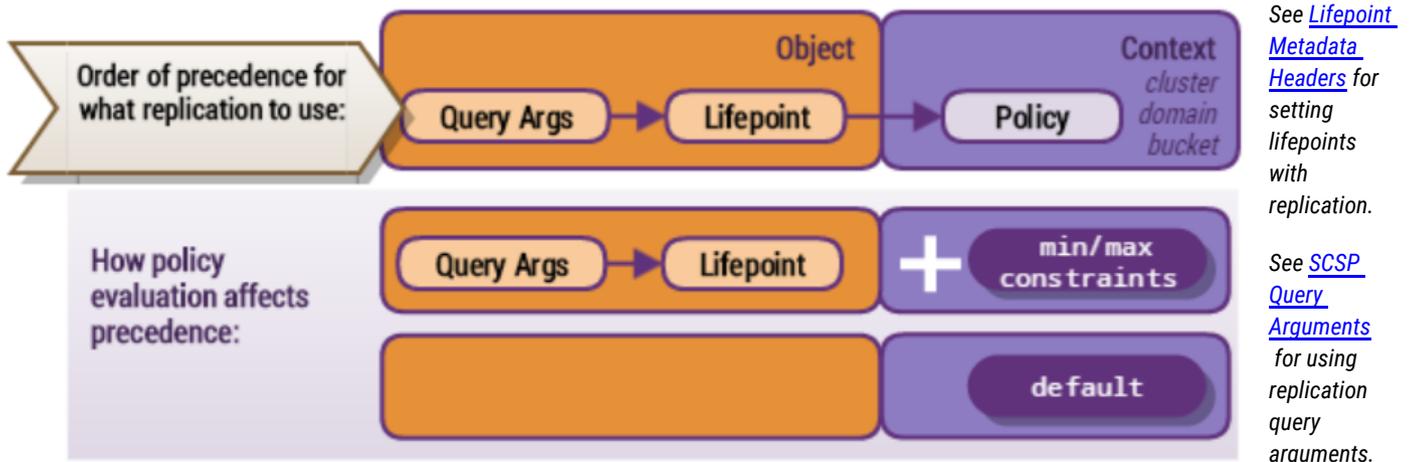
- [Implementing Replication Policy](#)

Implementing Replication Policy

- [Specifying Replication](#)
- [How Replication Policies Resolve](#)

Based on the context (location in the cluster) policies can be created to control how many copies of a replicated object Swarm maintains. Swarm allows maintaining separate replication policies for the cluster and any domain and bucket.

Understanding how Swarm chooses among policies is crucial to design policies because Swarm allows specifying different policies at multiple levels simultaneously. Replication specified at the object level have first priority, but the context policies constrain the minimum and maximum, as well as provide the default:



Specifying Replication

Replication is one of several content protection features in Swarm. All are controlled by way of *policies*. Swarm evaluates policies for each object based on the cluster and context values:

- Policy-related settings for the **cluster** (*required*)
- Policy-related headers on **domain and bucket** objects (*optional*)

For an overview of Swarm policies and how to set them at the cluster level, see [Configuring Cluster Policies](#).

The replication policy has three parameters. All are required at the cluster level because they establish the baseline for how the cluster replicates. To lock out lower-level policies apply the optional **anchored** parameter to cluster and domains.



Deprecated

The [cluster setting policy.replicas](#) replaces the following three, which are all [deprecated](#): `scsp.minReplicas`, `scsp.maxReplicas`, `scsp.defaultReplicas`



Note

Replication policy does not apply to context (bucket and domain) objects themselves, which follow the [cluster configuration settings](#) `scsp.defaultContextReplicas` and `scsp.maxContextReplicas`.

Using this 2-part setting define a different replication policy at the cluster and each context (domain and bucket) level:

Parts	Scope	Value	Notes
replicas required	Affects the <i>current</i> context level. Swarm returns 400 (Bad request) if value is missing or out of range. Replaces cluster-level settings: <ul style="list-style-type: none"> • <code>scsp.minReplicas</code> • <code>scsp.maxReplicas</code> • <code>scsp.defaultReplicas</code> 	<code>min:n</code> <code>1 n 16</code>	For the objects in this context, sets the lower limit for the number of replicas required to keep. Overrides any policy or lifepoint header <code>reps=</code> constraint of a lower value. <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Recommendations</p> <ul style="list-style-type: none"> • Use <code>min:1</code> and write those objects with <code>reps=1</code>, but leave the default <code>reps</code> at 2 or 3 to maintain single replicas of unimportant objects. • For <code>reps=3</code>, use the default <code>scsp.defaultROWAction = immediate</code>, unless the write load is heavy and sustained, in which case protect performance by using <code>scsp.defaultROWAction = full</code>. </div>

		<pre>max:n min n 20</pre>	<p>For the objects in this context, sets the upper limit for the number of replicas and for k+p for each EC/ECP write request.</p> <p>Overrides any policy or lifepoint header <code>reps=</code> constraint of a higher value.</p> <p>Small clusters: Set this value to the total number of nodes in the cluster, which avoids needless errors when lifepoint headers request more replicas than there are nodes.</p>
		<pre>default:n min n max</pre>	<p>For the objects in this context, sets the default number of replicas if none is specified by the current lifepoint or request.</p> <p>For most cases, <code>min</code> and <code>default</code> values match.</p>
<p>anchored</p> <p>optional</p>	<p>Affects contexts <i>below</i> the current.</p>	<p>none</p>	<p>Used with cluster and domain policies. Applies the policy to all levels below, overriding any lower-level policies.</p>

Summary of value constraints:

```
1 min default max 20
```

Example:

```
replicas min:2 default:2 max:6 anchored
```

Starting with the cluster set the replication policy for each level with the appropriate naming and method:

Scope	Type	Name, SNMP MIB	Configuration Method
<p>cluster</p>	<p>Configuration setting</p>	<pre>policy.replicas policyReplicas</pre>	<p><i>Cluster policies</i> are captured in a single setting including all three required parameters, for a baseline:</p> <pre>policy.replicas: min:2 max:16 default:2</pre> <p>Use the snmpset command on the cluster's settings object to change and persist the cluster-wide policies. See Using SNMP with Swarm.</p>
<p>domain and bucket</p>	<p>Persisted header</p>	<pre>Policy-Replicas policyReplicas</pre>	<p><i>Context (domain and bucket) policies</i> are captured in persisted headers overriding some or all cluster policies.</p> <p>Use a write method (POST for new, COPY for update) on the domain object to set the policy values.</p> <p>Use a read method (GET, HEAD) on the domain object to check the policy values.</p>

Replication is subject to policy resolution on each write request. For contexts, Swarm uses the policy at the given level unless a level above it is anchored. Swarm uses the anchored policy.

A new domain has a separate required policy for named objects (which bucket are unable to override). Tenanted unnamed objects follow this policy regardless of "anchored":

Writing a domain policy

```
curl -iL -XPOST --post301 --data-binary ""
-H "Policy-Replicas: max:6 default:4 anchored"
"http://{cluster}?domain=myDomain"
```

This new bucket increases the maximum, but note this policy is canceled by any policies "anchored" above it at the domain or cluster level:

Writing a bucket policy

```
curl -iL -XPOST --post301 --data-binary ""
-H "Policy-Replicas: max:9"
"http://{cluster}/myBucket?domain=myDomain"
```

How Replication Policies Resolve

The goal of policy resolution is to determine the correct number of replicas to keep for a content object if subject to a set of conflicting policies. This evaluation must be bounded by a minimum and maximum, and, as with all types of Swarm policies, must honor the **anchored** option, which inhibits evaluation at lower levels.

Swarm evaluates replication policy according to scope, which is tied to object type (because types can be stored in a certain type of context):

Scope	Object type	Min = <i>highest</i> among:	Max = <i>lowest</i> among:	Default = <i>first found</i> , constrained by min /max:
Cluster	<ul style="list-style-type: none"> • Untenanted unnamed 	<ul style="list-style-type: none"> • Cluster setting 	<ul style="list-style-type: none"> • Cluster setting 	<ol style="list-style-type: none"> 1. Lifepoint 2. Cluster setting
Domain	<ul style="list-style-type: none"> • Tenanted unnamed 	<ul style="list-style-type: none"> • Cluster setting • Domain object header 	<ul style="list-style-type: none"> • Cluster setting • Domain object header 	<ol style="list-style-type: none"> 1. Lifepoint 2. Domain header 3. Cluster setting
Bucket	<ul style="list-style-type: none"> • Named 	<ul style="list-style-type: none"> • Cluster setting • Domain object header • Bucket object header 	<ul style="list-style-type: none"> • Cluster setting • Domain object header • Bucket object header 	<ol style="list-style-type: none"> 1. Lifepoint 2. Bucket header 3. Domain header 4. Cluster setting

Every cluster update must supply all three values (min, max, default), but any subset is allowed for domain, bucket, and unnamed. They can be in any order, and any case.

Affected Objects	Example	Effect
------------------	---------	--------

cluster	Untenanted unnamed	<code>policy.replicas: min: 2 max:10 default:4</code>	Constrains the lifepoint value for <code>reps</code> : $2 \leq \text{reps} \leq 10$. Defaults to 4 if <code>reps</code> is unspecified.
domain	Tenanted unnamed	<code>Policy-Replicas: min: 1 max:8</code>	Constrains the lifepoint value for <code>reps</code> : $1 \leq \text{reps} \leq 8$. Defaults to 4 (from the cluster setting) if <code>reps</code> is unspecified.
bucket	Named	<code>Policy- Replicas: default:2 max:6</code>	Constrains the lifepoint value for <code>reps</code> : $1 \text{ (per domain setting)} \leq \text{reps} \leq 6$. Defaults to 2 if <code>reps</code> is unspecified.

Conflict Example

Policies at different levels can conflict, such as in this example:

First evaluation	= Invalid
Cluster: min:1 max:10 <ul style="list-style-type: none"> Domain: min:2 max:5 Bucket: min:6 max:8 	Most constrained values: <ul style="list-style-type: none"> min:6 max:5

The most restrictive min is 6, and the most restrictive max is 5, *which is not a valid pair*, because the max is lower than the min. Given such a conflict, Swarm logs a warning message and reevaluates the policy, omitting the lowest level in the hierarchy (bucket, in this example), until the conflict is resolved. The lowest level is dropped because Swarm privileges the values of the cluster owner over those of the domain owner, over those of the bucket owner.

First evaluation	= Invalid	Second evaluation	= Valid
Cluster: min:1 max:10 <ul style="list-style-type: none"> Domain: min:2 max:5 Bucket: min:6 max:8 	Most constrained values: <ul style="list-style-type: none"> min:6 max:5 	Cluster: min:1 max:10 <ul style="list-style-type: none"> Domain: min:2 max:5 Bucket: min:6 max:8 	Most constrained values: <ul style="list-style-type: none"> min:2 max:5



Note

Default values specified at lower levels override the default values set at higher levels, as long as they fall within the resolved min/max values.

Managing and Optimizing Feeds

- [Configuring Target Clusters](#)
- [Optimizing Replication Rate](#)
- [Deleting Search Data](#)
- [Feeds with Versioning](#)

Configuring Target Clusters

Uneven filling – Use one of these configuration strategies if there are concerns about uneven filling of the target (DR) cluster of the replication feed:

- Run DR clusters in full performance mode by disabling [Power-Saving Mode](#): `power.savingMode = false` (SNMP: `powerSavingMode`)
- Lower the setting that limits the difference in capacity between volumes, which defaults to 20%: `bidding.idleCost = 20` (SNMP: `biddingIdleCost`)
 - For pure DR clusters with no other traffic, set it to 5% to compensate for sleep cycles or feed definitions that favor particular nodes /volumes: `bidding.idleCost = 5`
 - For mixed-purpose clusters where remote replication causes uneven filling in the cluster, set it to 10%: `bidding.idleCost = 10`

Optimizing Replication Rate

You may need to adjust the rate at which replication occurs for these situations:

Need	Cause/Concern	Action
Speed up replication	Large volume of very small objects	Contact DataCore Support for specific settings adjustments.
Slow down replication	Low bandwidth and full cluster may trigger denial of service	In the networking routers/switches, enable the native rate-limiting features.

How Swarm parallelizes replication

The replication feed feature in Swarm seeks a high degree of parallelism in replicating objects between your source and target clusters. For each replication feed on each node, the replication feed is creating a batch of items to replicate. The size of the batch may be as large as 200 items, or smaller if a batch cannot be filled in 30 seconds. Once the batch of items has been filled sufficiently, it is sent to a node in the target cluster using long-running GET request that waits for the target cluster to replicate the items in the batch. When the batch of work has completed, the source cluster node can fill another batch. This mechanism creates a constant load of replication work for the target cluster. These GET/retrieve requests are relatively small and do not, in themselves, use much bandwidth.

Each node in the target cluster may be accepting work from multiple source cluster nodes from any number of source clusters and any number of feeds. Additionally, the source cluster may have a greater number of nodes than the target cluster. To prevent target cluster nodes from being overloaded, each node in the target cluster does two things. First, it delegates replication work to other nodes in the target cluster as a method of load balancing. Second, each node limits the number of replications that can be done simultaneously, regardless of source. Swarm's defaults assume a moderate client load.

Precise means of throttling can be achieved using networking technologies, including QOS bandwidth limiting and the use of bandwidth-limiting forward proxies out of the target cluster or reverse proxies into the source cluster.

Deleting Search Data

The Elasticsearch index (database) of search data remains on disk after you delete the feed; if you no longer need it, you need to delete it manually.

To delete the search data, you need to reference the search index, which is the same as the name of your cluster:

Delete search data

```
curl -X DELETE "http://{ip- elasticsearch}:9200/{cluster- name}"
```



Note

The Elasticsearch server manages additional indices related to Swarm cluster: Swarm Storage and Gateway store [historical metric](#) information in rolling indices. Deleting search data does not affect historical data.

Feeds with Versioning

Note: Swarm Versioning is supported for both feed types. Feeds process each object twice by default, on creation and deletion. Feeds push objects as frequently as needed to verify they stay current with versioning enabled.

For an introduction, see [Object Versioning](#) in *Swarm Concepts*.

For implementation, see [Implementing Versioning](#) in the *SCSP Reference*.

Replication Feed

Because object versioning is based on domains and buckets, which are replicated between clusters, object versions are also replicated between clusters. The replication feed processes historical object versions as well as current object versions.

Required

Upgrade both the source and target clusters to the same version of Swarm before enabling versioning in both clusters to use versioning with replications feeds.

Replication feeds replicate all versions, current and historical, to the remote cluster and allows the remote cluster to decide whether to keep the versions and how to integrate them into its version chain linkages.

Troubleshooting

It is possible for the older version to not be replicated in the target cluster if an object is versioned before the bucket/domain in the target cluster is updated to enable versioning. use the SCSP SEND command to re-transmit older versions manually if this occurs.

Search Feed

The search indices represent the current version of every object in the cluster. When a formerly obsolete named or aliased object becomes the current version again, the version number is based on an update time, provided on the object's metadata. This guarantees when Swarm decides a replica is the new current version, that fact is eventually updated in search. Because different replicas of the same object version may transition to "current" at different times, it is possible different replicas update the Elasticsearch record for an alias or named object more than one time. The latest such update wins. Swarm's update collision mechanisms prevent duplicate ES updates and minimize redundant updates.

Effect on search indices

- Two new indices represent *all* existing versions of aliases and named and the delete markers, with no information about which is the current version.
- When versioning is enabled or suspended, every new named or alias version creation results in a new versioned name or alias record.
- When specific versions are deleted, either by SCSP or by HP, the corresponding versioned record is removed.
- The primary key of these records is the version ID (Etag), which is unique to each version.
- Both alias and named object versions have a flag that indicates whether the version is a delete marker, and that information is captured in these indices.
- The search index schema upgrade for versioning does not require reindexing of your data.

Effect on searches

- Your existing search queries do not need to change.
- You can add the **versions** query argument to obtain all existing versions.
- Swarm returns versions in the order of the version chain, starting with the current version and ending with the original version. (When simultaneous updates occur, Swarm saves all updates but determines which position each occupies in the chain.)

Swarm Concepts

DataCore Swarm is unified storage software that leverages simple and emergent behavior with decentralized coordination to handle any rate, flow, or size of data. Swarm turns standard hardware into a scalable, highly available pool of storage resources that adapts to any workload or use case while offering a foundation for new data services.

This section introduces fundamental concepts to make full use of Swarm:

- How Swarm interacts with its content objects
- How you access objects and leverage feeds
- How you protect and secure your content
- How you store large objects efficiently
- How you use versioning in Swarm

- [Understanding Swarm Objects](#)
- [Working with Swarm Objects](#)
- [Version Control for Objects](#)
- [Understanding Feeds](#)
- [Working with Large Objects](#)
- [Elastic Content Protection](#)

Understanding Swarm Objects

- [Understanding Named Objects](#)
- [Types of Data Objects](#)
- [Types of Container Objects](#)
- [Understanding Unnamed Objects](#)

Understanding Named Objects

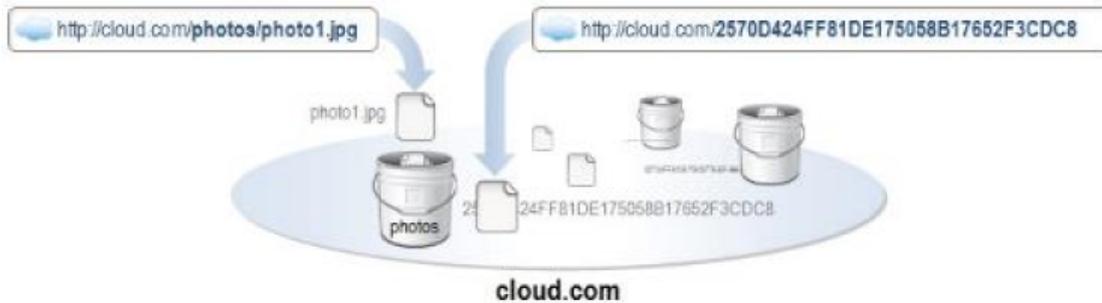
- [Named versus Unnamed](#)
- [Accessing Named Objects](#)
 - [Important](#)
 - [Note](#)
- [Creating Named Objects](#)
 - [Tip](#)
 - [Note](#)
- [Overwriting Named Objects](#)
- [Deleting Named Objects](#)
 - [Important](#)

Named objects provide a method to store and retrieve content by user-provided name, rather than by Swarm-assigned UUID. This structure makes it possible to use third-party protocols, such as S3.

A *named object* refers to a client-named object (such as a video file) contained within a bucket. Using meaningful bucket and object names creates friendly, readable URLs to stored content.

Named versus Unnamed

The illustration below contrasts two data objects—one named and one unnamed—in a cluster domain named `cloud.com`. The *named* object is stored in a bucket named `photos`, and the *unnamed* object is identified by a UUID at the root of the domain:



A Swarm private or public cloud can have whatever combination of object types needed:

- All unnamed objects
- All named objects (which must be assigned to buckets)
- Both named and unnamed objects in any proportion

Pathnames to named objects must be unique:

- Every object within a bucket must have a unique name.
- Every bucket within a domain must have a unique name.
- Two or more objects in the same cluster may have the same name, if they reside in different buckets or domains.

Accessing Named Objects

Access a named object by combining two strings: a **domain** name and a **pathname** to the object, in this form:

```
{domain-name}/{bucket-name}/{object-name/which/can/have/slashes}
```



Important

Even though objects in buckets resembles a file system, these are not files in folders. There is *no file system* underlying the apparent path structure of named objects.

`http://cluster.example.com/marketing/ads/about-object-naming.m4v`

In this example:

- **cluster.example.com** is the domain name.
- **marketing** is the bucket name. Buckets are required.
- **ads/about-object-naming.m4v** is the object name (**ads/** is part of the name).



Note

An object name like `/folder1/../folder2/object.txt` is *not* the same as `/folder2/object.txt`. These legitimate names specify *two different objects*.

Creating Named Objects

The following [cURL](#) examples detail creating a bucket and creating named objects within that bucket.

Tip

Use the [Swarm Content UI](#) to upload, view, and manage domains, buckets, and named objects from a browser.

Access to a domain without protection settings set to allow only a specific set of users POST privileges is required to use these examples. Create a domain with the following protection setting if access to a development environment is available:

All Users. No authentication required.

Contact the cluster administrator or see [Manually Renaming a Domain or Bucket in a Mirrored or DR Cluster](#) if unsure how to create a domain.

The following examples assume the domain is named `test.example.com` and commands are sent to a node with IP address is 172.16.0.35.

Open a terminal window and execute the following to create a bucket and objects:

1. Create a bucket. (replicate=immediate is the [Replicate On Write](#) option.)

```
curl -i --post301 --data-binary ''
  --location-trusted 'http://172.16.0.35/bucket?domain=test.example.com&replicate=immediate'
  -D create-bucket.log
```

2. Create a named object in the bucket.

```
curl -i --post301 --data-binary '<html><h1>Hello world</h1></html>'
  -H 'Content-type: text/html'
  --location-trusted 'http://172.16.0.35/bucket/test.html?domain=test.example.com'
  -D create-object.log
```

3. Verify the object in a browser. Enter the following in the *Address* or *Location* field:

```
http://172.16.0.35/bucket/test.html?domain=test.example.com
```

4. Change the object by adding data to it.

```
curl -i --post301 -X APPEND --data-binary '...to be continued...'
  -H 'Content-type: text/html'
  --location-trusted 'http://172.16.0.35/bucket/test.html?domain=test.example.com'
  -D update-object.log
```

5. Refresh the browser to see the updated object.

Note

Named and aliased objects can be updated at a maximum frequency of *once per second*. Updating more frequently can cause unpredictable results with the stored object version. Include the [replicate query argument](#) to verify more than one node can return the latest version in a subsequent read if an application updates objects faster than once per second.

Overwriting Named Objects

Overwrite a named object that currently exists in a storage cluster using the [POST](#) method and an HTTP request.

Include the **If-None-Match: *** request header to prevent overwriting an existing object.

- Swarm writes the named object if it does not exist.
- Swarm responds with an HTTP 412 Precondition Fail error if the named object exists.

See [SCSP WRITE](#).

Deleting Named Objects

Swarm allows deleting a bucket or domain that still contains content.



Important

Do not create orphaned content by deleting the associated container object.

Use one of these methods to avoid creating orphans:

- Delete the content first *before* deleting the bucket or domain.
- Add the [recursive query argument](#) when deleting the domain or bucket, which causes the health processor to remove asynchronously any content it finds in the deleted context.

Swarm may generate replication and search indexer warnings in the syslog and Admin Console as attempts are made to access content missing the parent context if one of the above methods is not performed. The logs include error messages displaying the alias UUIDs of the missing bucket or domain:

```
Domain 'example.com' (uuid=...) has been deleted with orphan content.
Consider recreating.
```

See [Restoring Domains and Buckets](#) for how to delete orphaned content.

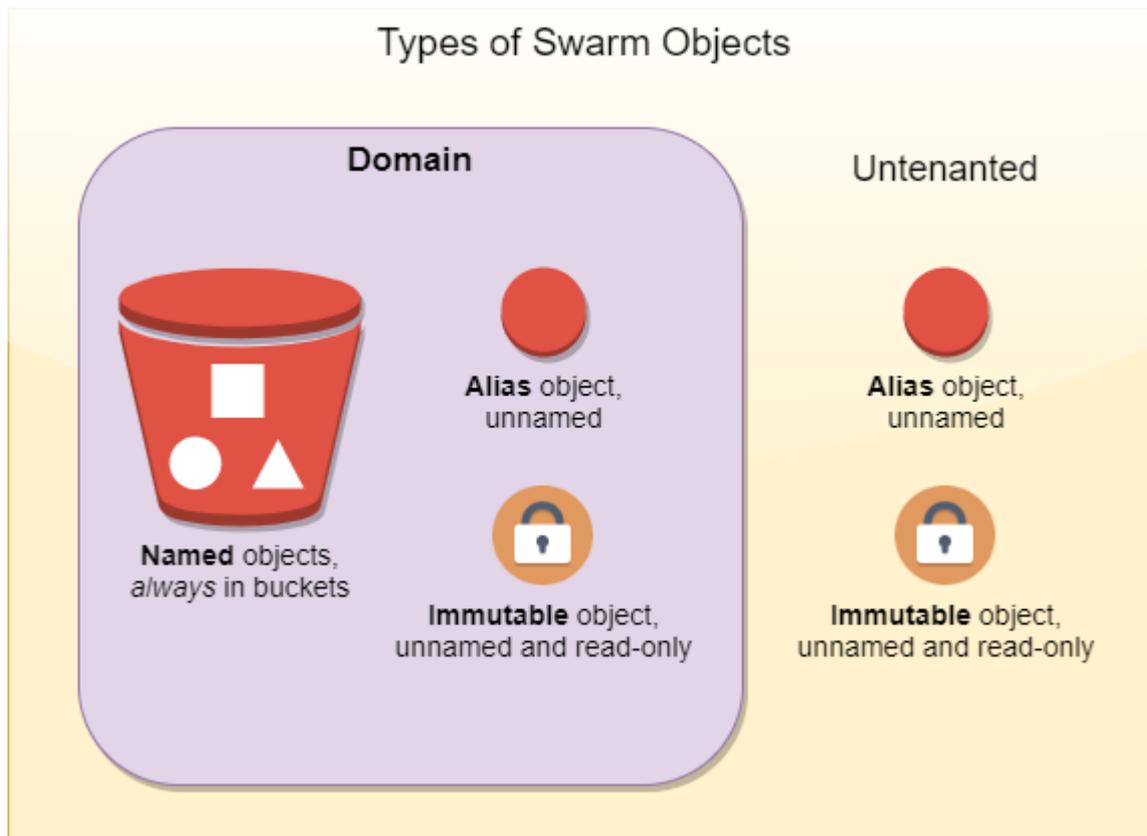
Types of Data Objects

- [Untenanted objects](#)

Data uploaded to Swarm is either *unnamed* or *named*:

- **Unnamed objects** are assigned identifiers by *Swarm*. They are created, updated, accessed, and deleted using an unambiguous UUID. Swarm is optimized to handle unnamed objects most efficiently, and the random assignment of UUIDs to content offers a higher level of programming security.
- **Named objects** are assigned identifiers by *the user*. They are created, updated, accessed, and deleted using the chosen name. Multiple objects can be created with the same name as long as they are in different *buckets*. Because these objects require a name lookup, they cannot perform as well as unnamed objects. Named objects are compatible with the AWS S3 protocol and Swarm clients interfacing with end users through a file system.

Swarm supports three types of data objects: **Named**, **Alias**, and **Immutable**. Two of the three can be updated (are *mutable*) and two of the three are identified by UUID (are *unnamed*).



Named objects

- Named objects must exist in buckets, which must exist in domains. A name only needs to be unique within a bucket. Named objects can be updated, and are always accessed by name and bucket. Another object can be created with the same name, in the same bucket if a named object is deleted.

Alias (mutable) unnamed objects

- Alias objects have permanent UUIDs but replaceable content. The UUID cannot be reused: *all* unnamed objects have the associated UUID retired after being deleted.

Immutable unnamed objects

- Immutable objects have UUIDs and content that do not change. Swarm retires the UUID of an immutable object when it is deleted. Using a special header, immutable objects can be locked to prevent deletion, allowing implementation of [Write Once Read Many \(WORM\)](#) (see [Lifepoint Metadata Headers](#)).



Untenanted objects

Earlier implementations of Swarm may have used *untenanted* unnamed objects, which are objects not tenanted in any domain.

- Untenanted objects are reached using the query argument "domain="
- Untenanted objects cannot be converted into tenanted objects and cannot be accessed via the [Content UI](#).

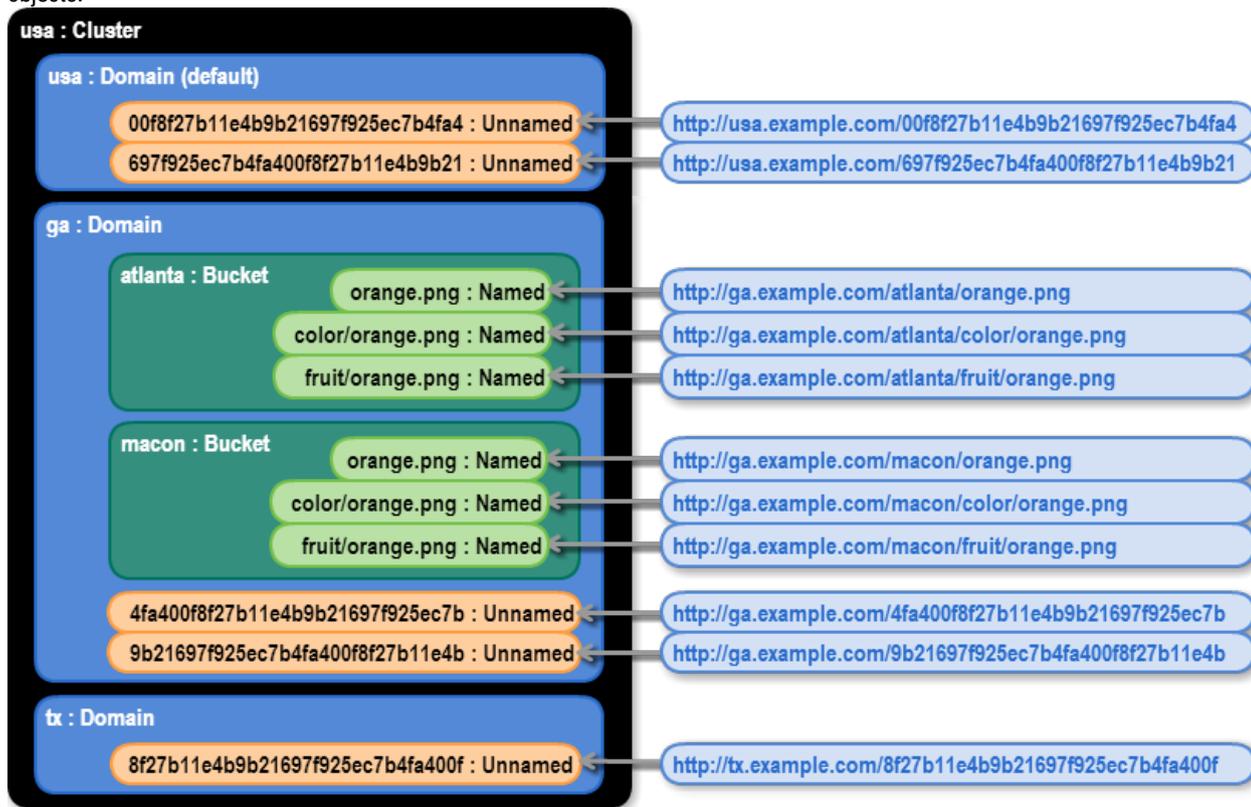
To manage objects with Swarm, applications communicate using the **Simple Content Storage Protocol (SCSP)** or else through the **S3 Protocol** (*named objects only*), via Gateway. SCSP methods and syntax are a subset of the HTTP/1.1 standard, with some extensions (see [SCSP Essentials](#)). The same SCSP methods apply to all types of objects.

See [Using SCSP](#).

Types of Container Objects

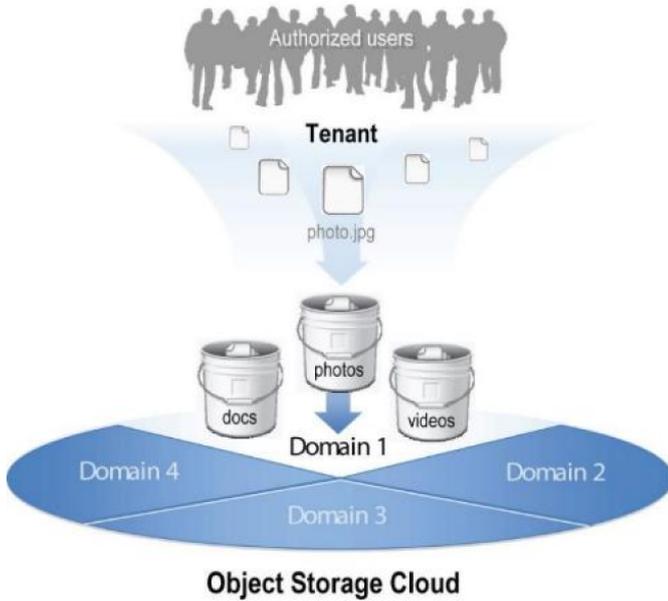
- [Clusters](#)
- [Domains](#)
 - [Best practices](#)
- [Buckets](#)
 - [Note](#)
 - [Important](#)

Swarm provides a hierarchy of **container** objects: *cluster*, *domain*, and *bucket* (shown in white, below) to facilitate organizing storage of data objects.



Clusters

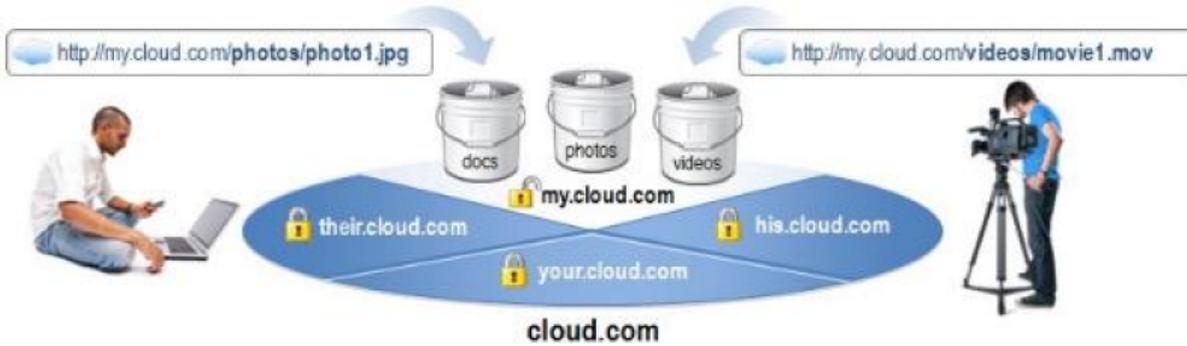
The *cluster* is the top-level container in an object storage cloud. A cluster can have *several* storage domains, each organized with a separate bucket:



These storage domains and buckets can enforce many business requirements:

- Assign permissions
- Issue metadata search or listing requests
- Account for usage and billing

Configure a cluster in to a storage cloud (such as `cloud.com`, below) that allows users in one storage domain (`my.cloud.com`) securely access content separately from other users:



Domains

A *domain* is the highest-level container in which data objects are placed, and it is the primary way to control access to and grouping of the data.

A cluster administrator creates the domain either using the Swarm Admin Console or [programmatically, with an SCSP command](#). Buckets and data objects must be created in a domain (*tenanted*), so plan to create domains first.

Best practices

These actions make URLs and cURL commands as simple as possible:

- Have DNS map the cluster name to one or more of the Swarm nodes.
- Create a storage domain that matches the name of the cluster. This is the default domain if no domain is specified in the SCSP request or if the specified domain does not exist.

See the [Naming Rules](#).

Buckets

Buckets are the required organizing containers for any named objects in a domain. Buckets allows creating logical paths to named objects, and they provide a layer of access control.

Each bucket operates as a subdirectory in the domain:

```
MYDOMAIN/my_bucket/my_document.doc
```

 **Note**

Buckets are not folders: they cannot be nested in other buckets.

The bucket must be created prior to storing a named object since named objects exist within buckets.

 **Important**

Do not write named objects without specifying a bucket (with a path like `http://cluster.example.com/testfile.txt`): performing this creates the object as a *bucket*. Buckets are treated differently and are not meant to contain a content body.

The cluster synchronously creates two replicas of the bucket object and then asynchronously creates any additional replicas as determined by a [lifepoint header](#) or the `scsp.defaultContextReplicas` and `scsp.maxContextReplicas` configuration parameters when creating a new bucket.

See the [Naming Rules](#).

Understanding Unnamed Objects

- [Untenanted objects](#)
- [Universally Unique Identifiers](#)
 - [Tip](#)
 - [UUIDs and Replicas](#)
- [Immutable Objects](#)
- [Unnamed Mutable \(Alias\) Objects](#)
 - [Update frequency](#)
- [Accessing Unnamed Objects](#)
 - [Note](#)
- [Creating Unnamed Objects](#)
 - [Caution](#)



Untenanted objects

Earlier implementations of Swarm may have used *untenanted* unnamed objects, which are objects not tenanted in any domain.

- Untenanted objects are reached using the query argument "domain="
- Untenanted objects cannot be converted into tenanted objects and cannot be accessed via the [Content UI](#).

Universally Unique Identifiers

Swarm assigns an unnamed object a unique identifier upon creation that is different from every other identifier assigned to every other unnamed object ever stored, past and future. This identifier is known as the object's *Universally Unique Identifier* (UUID).



Tip

Think of a UUID like a coat check ticket. When checking a coat at a restaurant, a coat check ticket identifies the coat owner. To retrieve the coat when ready to leave, present the ticket, not a description of the coat. No ticket, no coat.

A Swarm UUID is a sequence of 128 bits. In text-based languages and protocols such as Swarm's [Simple Content Storage Protocol](#), a UUID is represented as a sequence of 32 hexadecimal digits:

```
http://companyname.example.com/12BFEA648C2697A56FD5618CAE15D5CA
```

A UUID has no internal structure and cannot be parsed in any way to yield information about where or when the associated object was stored. *An object's UUID is not derived from its content.*

UUIDs and Replicas

Swarm produces copies (or *replicas*) of an object to facilitate fault-tolerance, integrity, or speed of access. Every replica of a given object has exactly the same UUID as the original object. There is no external way to distinguish an original from a replica. Consider each replica of an object completely identical to every other replica of that object in the cluster.

Immutable Objects

By default, an unnamed object changes only by deleting it (if permitted by the object's lifepoint policy) once the object is stored in Swarm. There is no way to open an immutable object for updates or appends to the content or metadata. It is possible to decorate an immutable object with metadata that is stored separately in an annotation; see [Metadata Annotation](#).

To store an object in the cluster, an application must provide a size (in bytes) for the object. Swarm then allocates exactly enough space to store the given number of bytes. After that, every object replica has precisely the same byte size. There is no possibility a replica is updated or changed in some way while others are not changed.

If an application needs to update an immutable object with new content, the application is responsible for:

- Storing a new Swarm object containing the updated data (or delete it if necessary)
- Modifying any references to the old UUID to point to the new UUID
- Maintaining association between the old object and the new revision

If those associations are important for a situation, consider using alias objects.

Unnamed Mutable (Alias) Objects

Alias objects are a special type of unnamed object in Swarm because the content can be replaced and the UUID remains constant (or *alias*). An alias object is created in much the same way as a regular object but uses an **?alias** query argument on the **WRITE** request. Unlike an immutable object, where the contents do not change, the contents of an alias object can be replaced using an SCSP **COPY**, **APPEND**, or **UPDATE** request. Swarm always returns the most recent data associated with the alias object object's UUID when reading an alias object.

Alias objects serve a very specific purpose for applications that store fixed content data. Many such applications must associate a symbolic name of some kind (e.g. a URI or a file pathname) to an object UUID returned from Swarm.

Identifying an alias object – Identify an alias object by examining the metadata:

- An alias object's UUID is different from the ETag value.
- An alias object may also have a `castor-system-alias` metadata entry as well



Update frequency

Named and alias objects can be updated at a maximum frequency of *once per second*. Updating more frequently can cause unpredictable results with the stored object version. If an application updates objects faster than once per second, include the [?replicate query argument](#) to guarantee more than one node can return the latest version in a subsequent read.

Accessing Unnamed Objects

Access unnamed objects using the UUID, placing it at the end of the URI as a string of 32 case-insensitive hexadecimal digits:

```
http://companyname.example.com/12BFEA648C2697A56FD5618CAE15D5CA
```

This URI specifies three things:

- **Protocol:** `http`
- **Cluster:** `companyname.example.com`
- **UUID:** `12BFEA648C2697A56FD5618CAE15D5CA`



Note

The length of the UUID string must be *exactly* 32 characters, including any leading zeros.

Creating Unnamed Objects

The UUID is not assigned when storing an object for the first time, so an [SCSP WRITE](#) request includes the first two components of the URI. Swarm generates and returns a new UUID to the storing application after the data is transferred and stored.

For how the UUID is returned, see [Normal Responses to WRITE](#).

Every request must include one of the following:

- a `HOST` header equivalent to the cluster name
- the host IP address
- a `domain=clusterName` query argument



Caution

Verify applications are not passing a `HOST` header as neither an IP address nor a domain existing in the cluster (unless the host header matches the cluster name). Swarm attempts to look up the non-existent domain on every request and waits for multiple retries before the lookup times out, impacting performance.

Best practice: Add the `domain=<domain-name>` query argument, which overrides this lookup and prevents the timeout penalty.

Working with Swarm Objects

- [Content Cache](#)
- [Naming Rules](#)
- [Universal Resource Identifiers](#)
- [Data Protection](#)

Content Cache

Swarm uses a content cache to store frequently accessed objects (primarily domains and buckets). These objects can be cached in RAM on the requested nodes, which boosts read throughput and response times for relatively small objects that are accessed frequently.

Important

To maintain performance, do not disable the content cache unless advised by Support, especially if you are writing named objects.

As object demand changes over time, Swarm automatically manages the cache to increase or decrease the number of cached copies throughout the cluster. For most objects, this prevents stale data being returned in a query.

These cluster-wide parameters configure the cache:

- `cache.expirationTime`
- `cache.maxCacheableSize`
- `cache.percentage`
- `cache.realmStaleTimeout`

See the [Settings Reference](#) for configuring the content cache.

See [Use the Content Cache in a Distributed System](#).

Response headers for the content cache

The following response headers provide information about the content cache:

- **Age.** Indicates the length of time (in seconds) the object was stored in the content cache.
 - If the Age header is absent, the object was retrieved from the drive.
 - If the Age header = 1, the object is cached on a node where it also resides on the drive. See [RFC 7234 5.1](#).
- **Cache-Control: no-cache.** Matches exactly what was sent with the object on WRITE.
- **Cache-Control: max-age.** Matches exactly what was sent with the object on WRITE.
- **Cache-Control: no-cache-context.** Matches exactly what was sent with the object on WRITE.

See [Caching Metadata Headers](#).

Naming Rules

- [Slashes](#)
- [Note](#)

Follow these rules for naming the domain, bucket, and named objects created for storage in Swarm.

Slashes

Swarm handles slashes this way (v11.1):

- Leading slashes (/foo) are silently removed in all cases.
- Trailing slashes (foo/) are silently removed for buckets, but they cause *404 Page Not Found* errors for domains.
- Trailing slashes (foo/bar/) are preserved for object names because they are valid.

Type	Reference	Rules and Notes	Examples
Tenant	RFC 1034	<p><i>Applies to Gateway only.</i></p> <p>A tenant must follow the naming rules of a domain.</p>	
Domain	RFC 1034	<p>For maximum compatibility, verify the domains are valid DNS names that resolve in your network.</p> <p>A domain name must</p> <ul style="list-style-type: none"> • Be a 7-bit ASCII byte sequence. • Be case-insensitive. • Begin with an alphanumeric character. • Use alphanumeric characters, underscore (_), period (.), and hyphen (-). • Not have adjacent or final hyphens or periods (-, .., -,.-). • Not be an IPv4 or IPv6 IP address. • (S3 compatibility) Not be longer than 253 characters. 	<p>Valid:</p> <pre>my-cluster.example.com my_cluster.example.com</pre> <p>Invalid:</p> <pre>domain cluster_example_com</pre>
Bucket	RFC 1034	<p>A bucket name (which is only used in the path) must</p> <ul style="list-style-type: none"> • Be unique within the domain. • Be case-sensitive. • Be a valid URL-encoded, UTF-8 byte sequence. <ul style="list-style-type: none"> • <i>Content UI:</i> URL encoding is managed by the user interface. • Not be a UUID (32 hexadecimal characters). • Not exceed 8000 characters (greater than that is not tested or supported). • (S3 compatibility) Use lowercase ASCII and DNS-compatible names not longer than 63 characters. 	

Named object	RFC 3986	An object name must <ul style="list-style-type: none"> • Be unique within the bucket. • Be case-sensitive. • Be a valid URL-encoded, UTF-8 byte sequence. <ul style="list-style-type: none"> • <i>Content UI</i>: URL encoding is managed by the user interface. 	Valid: <code>Accounting/</code> <code>Customer23-03/15</code>
---------------------	--------------------------	---	--


Note

While you may use non-ASCII characters (such as "résumé.doc") in bucket and object names, the URL must be properly escaped in the HTTP request ("r%C3%A9sum%C3%A9.doc").

Universal Resource Identifiers

- [Supported Application Protocols](#)
- [Addressing a Cluster](#)
 - [Important](#)

A [Uniform Resource Identifier](#) (or *URI*) is a string of characters that identifies a resource over the network. The character string can be organized in a logical format (such as an object name) or a system-generated set of characters organized in a random pattern (such as a UUID):

```
http://companyname.example.com/12BFEA648C2697A56FD5618CAE15D5CA
```

The object's name or UUID must be known to identify the object to any storage cluster holding at least one replica of the object.

Address the cluster and specify a protocol delivering the data to retrieve an object over a network. A URI as defined in [RFC 2396](#) allows specifying these components in a compact form:

- Protocol (`http`)
- Cluster name (`companyname.example.com`)
- Object name or UUID (`12BFEA648C2697A56FD5618CAE15D5CA`)

Supported Application Protocols

Swarm only understands the application-level protocol called [SCSP](#), which is a subset of the Hypertext Transfer Protocol ([HTTP/1.1](#)) used by web servers and browsers. While additional access protocols may be added to Swarm in the future, HTTP is the only protocol supported direct to Swarm at this time.

For how to access Swarm via S3, see [S3 Protocol Interface](#).

Addressing a Cluster

Address a storage cluster using either:

- The DNS name
- The IP address of a cluster node

Which node selected is not important, as long as the node is accessible to the application on the network. The node named in the URI is *not* required to be the same node that initially stored the object because any node can be asked to retrieve any object stored on any node in the cluster.



Important

It may be required to change the URI used if the addressed node is down or off-line.

Data Protection

Swarm is designed to provide both fast and *secure* data storage. Selecting the appropriate trade-offs between performance, capacity, and data protection that best meet your business needs is a critical part of a successful storage solution.

Role of the Health Processor

Swarm includes the Health Processor (HP) that provides end-to-end, disk-level, and life-cycle data protection. The Health Processor monitors both data integrity and cardinality (the number of object replicas) continuously, and it heals any degradation or non-conformity within the storage cluster that it finds.

The Health Processor regularly scans every object on disk to verify its integrity. If the object is corrupt (such as due to a bad sector), HP removes it. HP then detects and corrects the object's under-replication by triggering creation of another replica of that object elsewhere in the cluster, thus restoring the correct number of replicas.

Using the Health Processor, Swarm can:

- Verify the correct object protection level exists in the cluster.
- Distribute replicas and erasure coding segments properly across subclusters.
- Enforce lifepoint policies by enforcing replica and/or erasure coding segment counts and delete policies (*terminal* lifepoints) at different policy time intervals.
- Validate the object on-disk integrity and create a new replica if the integrity is compromised.
- Economically store and load balance by periodically evaluating if an object is optimally stored in its current location.
- Defragment storage space on an as-needed basis.

See [Lifepoint Metadata Headers](#).

Content-MD5 Integrity Checking

In addition to the protection provided by the Health Processor, Swarm uses the **Content-MD5** metadata header to provide end-to-end message integrity check of the object body (excluding metadata) as it is sent to and from Swarm.

See [Content-MD5 Checksums](#).

Version Control for Objects

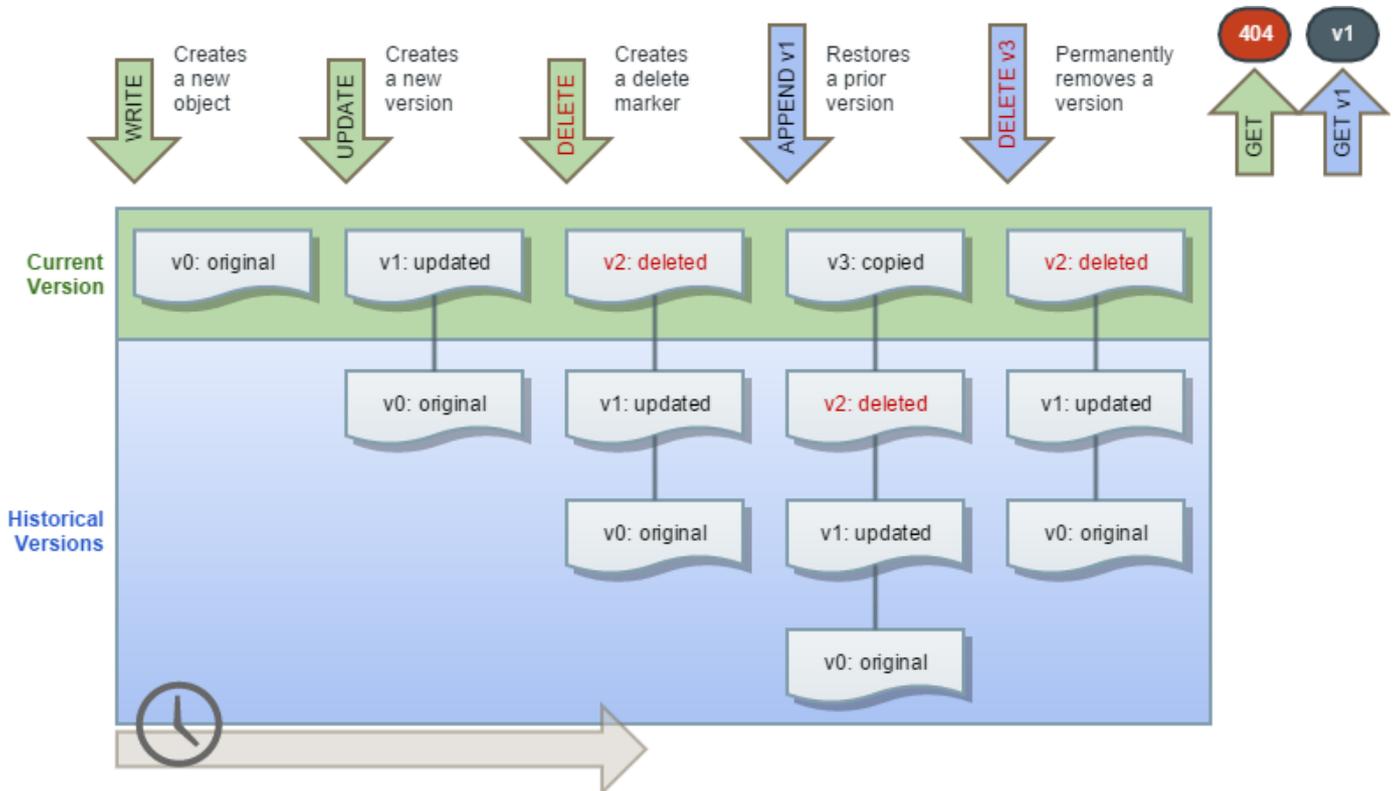
- [Note](#)
- [S3 Versioning](#)
- [Why use versioning?](#)
- [What is versioned?](#)

Object-level versioning is a powerful content protection option that tracks, secures, and provides access to historical versions of objects, even after they are deleted. With versioning, applications can read, list, revert, and purge prior versions as well as restore objects deleted by mistake.

Note

Using Swarm versioning with SCSP operations has no dependencies. To use Swarm versioning with [Amazon S3](#), Content Gateway version 4.1 or higher must be run.

Versioning preserves a set of historical variants of an object, the original plus subsequent updates to it, up to and including the latest version:



These are key capabilities of Swarm versioning:

- **Unlimited versions** – The number of supported versions for a given object is unbounded, and all versions have a unique version ID. List all versions, access, restore, and permanently delete specific versions via the version ID.

- **Flexible policy** – The cluster administrator changes the cluster policy settings to allow versioning; the domain administrator can then allow and even require versioning in that domain. A bucket owner can enable/disable versioning for a specific bucket if allowed by the cluster and domain.
- **Lossless concurrent updates** – Swarm captures simultaneous PUT updates and resolves the order in the version chain. Swarm preserves all versions, even those overlapping in time, with the latest update as the current version.
- **Accurate disk reporting** – Each object revision in a domain/bucket with versioning-enabled preserves and reports the full size on disk. Swarm includes all object revisions in the 'du' responses if requested. The size for deleted and historical versions counts towards bucket and domain totals.
- **Support for search and replication** – Swarm Versioning works with both [Search feeds](#) and [Replication feeds](#), provided all clusters are running the same version of Swarm.

S3 Versioning

Swarm's native object versioning feature is interoperable with AWS S3 versioning. The implementation includes these improvements:

- *Ability to disable versioning:*
AWS S3 allows for versioning to be suspended once enabled on a bucket. Swarm provides the ability to disable versioning and automatically clean up the prior versions to reclaim storage space.
- *Delete marker consolidation:*
Unlike AWS S3 where continued DELETE operations on a deleted object record additional delete markers in the version history, Swarm acknowledges the subsequent deletes without recording additional delete markers. Multi-factor authentication delete is not supported.
- *Expanded version listing:*
Swarm supports version listing batches up to 2000 items while AWS S3 limits these listing results to batches of 1000. Additionally, Swarm does not break batches on version boundaries. Delimiter case is currently not supported for version listing.
- *Simplified ACL management:*
When using per-object ACLs with versioning, the ACL for the current version of the object applies for determining authorization. To change the ACL for an object's entire version chain, update the object *without* specifying a version.

Why use versioning?

Versioning meets two key needs:

- Require extremely durable data retention and archiving.
- Require the ability to recover when data is erroneously overwritten or deleted.

With versioning enabled, prior versions of a stored object, can be retrieved and restored allowing recovery from data loss, whether caused by user error or application failure:

- **Deleting an object** – Swarm inserts a delete marker instead of removing it permanently, which becomes the current object version. Previous versions can be restored.
- **Overwriting an object** – Swarm performs the update by creating a new version, allowing restoring previous versions by rolling back a bad update.

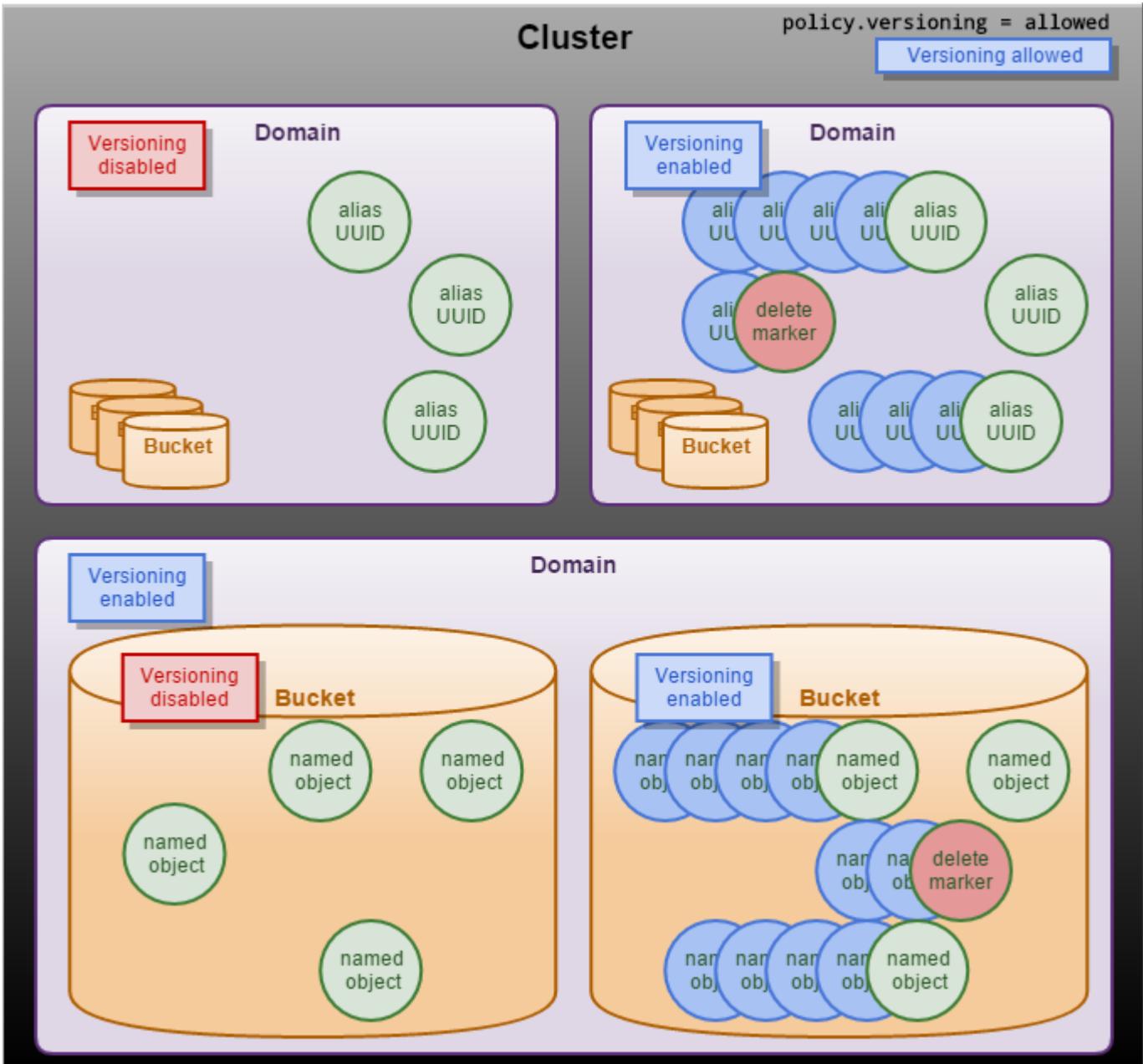
By default, versioning is disabled across the cluster. To avoid excessive storage usage, enable versioning in a targeted way, where change control is required.

What is versioned?

Choosing to use versioning enables the ability to preserve, retrieve, and restore every update of every object stored in that context (domain or bucket). With Versioning, Swarm archives another copy of an existing object when an update or delete is processed. GET requests retrieve the most recently written version, but retrieval of older versions of an object is performed by specifying a version in the request.

Administrators can selectively enable versioning at the following levels once the **cluster** is configured to allow versioning:

- the **domain** level (for *alias* objects)
- the **bucket** level (for *named* objects)



When a DELETE operation is issued against a versioned object, Swarm creates a delete marker so subsequent unversioned requests no longer retrieve the object. Swarm stores all versions of an object so they can be retrieved and restored.

Note: these types of Swarm objects cannot be versioned:

- Domains
- Buckets
- Unnamed objects (which are immutable)

- Alias objects not tenanted in a domain

Objects are versioned while domains and buckets are not (contexts). A bucket *is* lost if accidentally deleted. Swarm pauses the recursive delete of the bucket's contents for the duration of the grace period ([health.recursiveDeleteDelay](#)). There is time to recreate the bucket with the same headers to avoid data loss (see [Restoring Domains and Buckets](#)). The content starts to disappear as Swarm's Health Processor begins cleaning up all versions of the obsolete content, to reclaim space, if the bucket is not restored and the grace period expires.

- [Versioning Examples](#)
- [Working with Versioning](#)
- [Implementing Versioning](#)
- [Versioning Operations](#)

Understanding Feeds

Feeds is the object-routing mechanism in the Swarm storage cluster that uses intermittent channel connections to distribute data to these targets:

- *Elasticsearch cluster*, for object metadata search
- *Remote Swarm storage clusters*, for object replication and mirroring
- *S3 cloud service bucket*, for object replication dedicated to disaster recovery

Swarm supports three types of feeds for implementation:

- **Metadata Search** (*recommended, and required for Content Gateway and UI*) provides real-time metadata indexing and ad-hoc search capabilities within Swarm by name or metadata. The Elasticsearch service collects the metadata for each object and updates the search database in your Swarm network. When you create a new object, domain, or bucket, the service collects *only the metadata and not the actual content*.
See [Search Feeds](#).
- **Remote Replication** enables object replication directly to an external storage cluster without API intervention. When Swarm recognizes new or updated objects, it copies these objects to an internal queue. At specific intervals based on the `retryWait` attribute settings, the plug-in moves the queued objects to the targeted cluster.
See [Replication Feeds](#).
- **S3 Backup** provides off-premises Swarm disaster recovery from an S3 cloud service. It allows backup and selective restoration of some or all Swarm cluster contents.
See [S3 Backup Feeds](#).

Because feeds involve communication beyond the Swarm cluster, they require a reliable network connection between the source and target cluster, as well as the source cluster and the Elasticsearch cluster.

Search feeds work on the entire cluster, but replication-type feeds have optional domain filtering (inclusive and exclusive). They can copy over to the target all content in the source cluster or the content in specific domains.

Feed Behaviors

- Feeds operate continuously to keep up with source cluster changes.
- A single object can be associated with up to *eight* feed definitions.
- The source cluster processes all UUIDs and names stored in the source cluster based on your feed configurations. As objects are added to the cluster, Swarm adds the UUIDs and names to the assigned feed queue. Swarm logic processes the queue and notifies the target cluster and the Elasticsearch server that feeds are available.
- All feed changes can take up to 60 seconds to propagate from the source node to the targeted nodes in your cluster.
- After an object is replicated, it is not replicated again unless you update a feed. When this occurs, Swarm reevaluates all objects in the source cluster against the new feed definition. If required, Swarm reinitiates another replication to the targeted cluster.



Tip

To determine whether a particular object was replicated or indexed for metadata search, use administrator credentials to submit an [SCSP INFO](#) request for the object and view feed status information in the metadata for the object.

Working with Large Objects

- [Note](#)
- [Dividing Objects with Erasure Coding](#)
- [Storing Large Objects](#)
 - [Increasing allowed object size](#)
- [Storing Streaming Media](#)

To work with very large objects or objects of unknown length, you need to use the advanced options that are incorporated in Swarm Elastic Content Protection:

- [Erasure coding](#) (EC), which segments and stores large objects efficiently and securely
- [Multipart Write](#), which divides an object into multiple parts and uploads them simultaneously

These are key terms used in Swarm elastic content protection:

<p>Chunked transfer encoding</p>	<p>Used in WRITE, UPDATE, and APPEND SCSP methods to send objects of an undetermined content length to a storage cluster.</p> <p>The exact request header is:</p> <pre>Transfer-Encoding: chunked.</pre> <p>See RFC 7230 3.3.1.</p> <div data-bbox="392 1056 1484 1228" style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>COPY rewrites the object manifest only.</p> </div>
<p>Erasure coding</p>	<p>Describes one of the ways an object can be protected in a storage cluster.</p> <p>A large object written to the cluster using erasure coding is automatically stored on disk as a set of data and parity segments. This process verifies both content protection and optimal storage usage for large objects. Swarm has configuration parameters that enable an object to be automatically erasure-coded on the drive.</p>
<p>Manifest</p>	<p>Swarm object containing a list of the segments that comprise a large object.</p>

Dividing Objects with Erasure Coding

Swarm allows writing large objects of known length using the erasure coding option incorporated in the Swarm Elastic Content Protection. With this option, you can divide the object into smaller segments and encode it with additional parity segments that provide data protection.

Additionally, you can write ([POST](#), [PUT](#), [COPY](#), [APPEND](#)) objects of unknown length to a cluster using standard [HTTP chunked transfer encoding](#). Objects sent to the cluster using chunked transfer encoding are erasure-coded when stored on disk, using the encoding type specified by either cluster configuration or request query arguments. This feature allows you to store large objects and streaming media in the cluster.

Storing Large Objects

You can store an object as large as 4TB in the cluster. Erasure coding is seamless and transparent to the application, automatically partitioning the object into *segments*, encoding them, and distributing the segments throughout the cluster. When you configure the cluster, you set the threshold for when objects become erasure coded; in addition, applications can control which objects get erasure-coded on an individual object basis. See [Erasure Coding EC](#).

Attempting to store an object larger than 4TB results in an HTTP 400 Bad Request response immediately after the write is submitted.



Increasing allowed object size

To store objects *larger* than 4TB, increase the limit that is set by **ec.maxSupported** (defaults to 4398046511104) and also set **ec.segmentSize** (defaults to 200000000) to a value proportionately larger. On a full read, Swarm must load the entire manifest; increasing the segment size minimizes the size of the manifest and so the number of socket connections required to read an entire EC object. (SWAR-7823)

Storing Streaming Media

Streaming media is supported using industry-standard chunked transfer encoding. Your application can now stream digital media or other types of data to the cluster without knowing the object size in advance. The size of the object is limited only by the available space in the cluster (up to 4TB). Attempting to store a chunked encoded object larger than 4TB results in an HTTP 400 Bad Request response (see note above).

Any object written with HTTP chunked transfer encoding must be erasure-coded and cannot be replicated. If you write an object and specify both erasure coding and replication in the header (for example, combining an `encoding=5:2` query argument with a lifepoint header with a `reps=` parameter), the write operation results in an HTTP 400 Bad Request response.

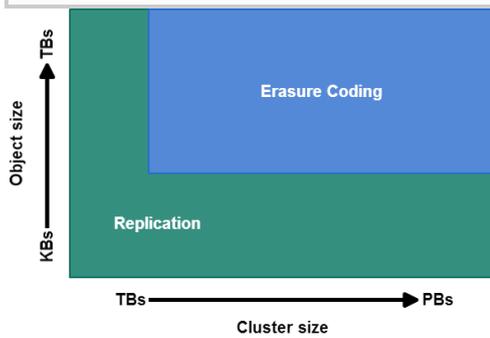
Elastic Content Protection

Swarm allows you to flexibly determine the type and level of content protection that best fits your storage needs using Elastic Content Protection. Objects can be either replicated or erasure-coded, with objects of both types co-existing in the same cluster.



Tip

Erasure coding is best suited for clusters with many nodes and larger objects, while replication is advantageous in smaller clusters and with smaller objects.



Content Protection with Replication

- [Note](#)
- [How Replication Affects Risk](#)
- [Controlling Replication Protection](#)
 - [Caution](#)
 - [Deprecated](#)
- [Increasing Replication Priority](#)
- [Enforcing Replicate On Write \(ROW\)](#)

Swarm can provide protection on disk by creating multiple copies of each object on different nodes called *replicas*. Control how many replicas are created for each object and how quickly they are created after the object is initially stored in the cluster.

Note

There is one instance of an object in the cluster if one object replica exists in a cluster. In this context, *replica*, *instance*, and *object* are all synonymous.

How Replication Affects Risk

By default, each object in Swarm is stored with two replicas, with each replica residing on a different node in the cluster. Replicas are distributed across subclusters if a cluster is configured to use subclusters.

In the event of a total failure or a hard drive fails for any reason, the cluster reacts quickly and initiates a volume recovery process for each missing drive. The recovery process rapidly creates additional replicas elsewhere in the cluster of all objects stored on the now missing drive(s) so each object again has two replicas. There is a protection risk for the sole replica of the object in the cluster if a second drive fails before the recovery process is completed.

There can also be a potential period of vulnerability at the moment an object is first stored on Swarm if Replicate On Write option to create multiple *simultaneous* replicas is not used.

Controlling Replication Protection

A rapid sequence of drives to fail is possible, but unlikely. The solution is to change the replication requirements if this presents an unacceptable risk for an application. Changing the default replication requirements to a greater number of replicas allows a trade of disk space savings for added security.

Caution

Specifying too many replicas relative to nodes has consequences. Setting the number of replicas equal to the number of storage nodes can lead to uneven loading when responding to volume recoveries.

To set the replication protection for the cluster, configure a single setting, `policy.replicas`, with three required parameters, for `min`, `max`, and `default` number of replicas:

```
policy.replicas: min=2 max=5 default=3
```

Deprecated

The [cluster setting policy.replicas](#) replaces the following three, which are all [deprecated](#): `scsp.minReplicas` `scsp.maxReplicas` `scsp.defaultReplicas`

See [Implementing Replication Policy](#).

Increasing Replication Priority

By default, Swarm writes a new object to one node, responds to the application with a success code and UUID (or name), and then quickly replicates the object as needed to other nodes or subclusters. The replication step is performed as a lower priority task.

While this creates the best balance of throughput and fault tolerance in most circumstances, there are cases where you may want to provide the replication task the same priority as reads and writes, which guarantees replication occurs quickly even under heavy sustained loads.

The cluster administrator can add the following parameter to the node or cluster configuration file:

```
health.replicationPriority = 1
```

With replication set to priority 1, object replication is interleaved in parallel with other operations. This may have a negative impact on cluster throughput for use cases involving sustained, heavy writes. With `health.replicationPriority = 1`, it is still possible (though much less likely) that the failure of a node or volume can cause some recently written objects to be lost if the failure occurs immediately after a write operation but before replication to another node can be completed.

Enforcing Replicate On Write (ROW)

Another replication strategy to protect content is *Replicate on Write* (ROW).

Without ROW, the client writes a single copy and depends on the Health Processor (HP) to create the necessary replicas. Relying on HP leaves open a small window for data loss: the volume containing the node holding the sole copy can fail before HP completes replication. ROW eliminates the window by guaranteeing all replicas are written on the initial request.

How it works: The ROW feature requires Swarm to create replicas in parallel before it returns a success response to the client. ROW protection applies to `WRITE`, `UPDATE`, `COPY`, and `APPEND` requests. The secondary access node (SAN) sets up connections to the number of available peers required to create the needed replicas when ROW is enabled.

See [Configuring Replicate On Write](#).

- [Implementing Replication Policy](#)

Content Protection with Erasure Coding

Replication is a proven and valuable mechanism to verify data integrity, but the cost per GB of storage can become high as object sizes and cluster sizes expand. A complementary data protection strategy, erasure coding (EC), provides high data durability with a smaller footprint. Swarm manages EC and replication together to optimize cost-effectiveness, converting objects between them seamlessly and dynamically, based on the policies set.

- [How EC works](#)
- [How much EC protects](#)
- [How much disk space EC saves](#)
 - [Note](#)

How EC works

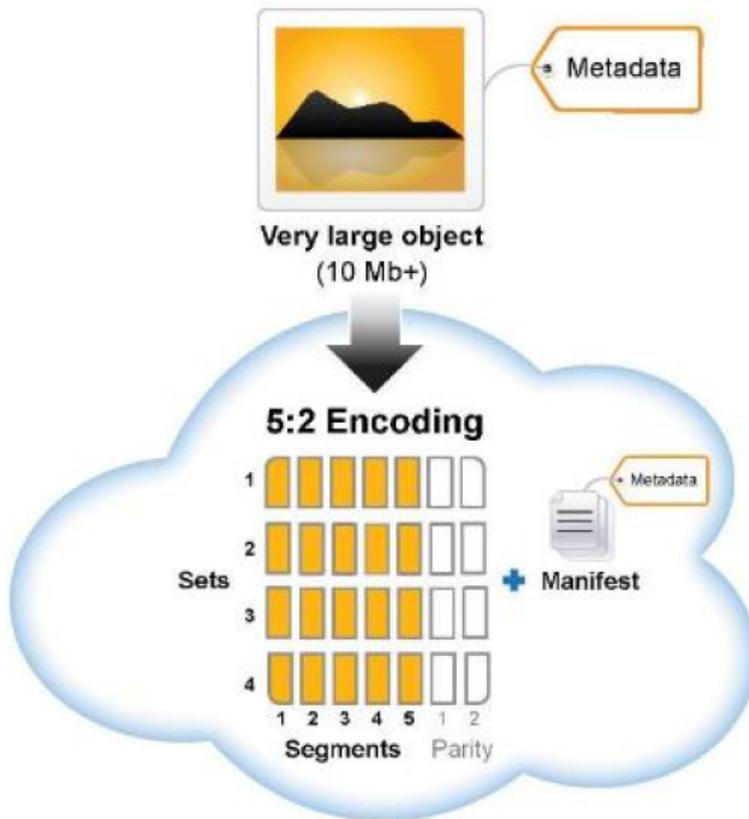
Erasure coding breaks the original object into multiple **data segments (k)** and computes additional **parity segments (p)** based on the content of the data segments. This results in m total segments ($k + p = m$) being distributed to m different nodes (or subclusters) in the storage cluster (see the [ec.protectionLevel](#) setting).

The erasure coding encoding level is expressed as a tuple in this format:

```
{data segments}:{parity segments}
```

Swarm creates multiple sets of erasure segments for *very* large objects. The object breakdown into one or more erasure sets is transparent to external applications. A GET or HEAD of an erasure-coded object uses the same syntax as a replicated object.

The following illustration represents how erasure coding works:



How much EC protects

Swarm still reads the object as long as there are still k total segments (any combination of original data or parity) remaining in the cluster if a hard disk or a node containing an erasure segment fails. Protection against disk failure for the object is equal to the number of specified parity p segments.

The segments from a 5:2 (5 data segments with 2 parity segments for a total of 7 segments) or 8:2 (8 data and 2 parity segments for 10 total segments) erasure code are protected against the loss of any two nodes because they are distributed to different nodes. An erasure-coded object is immediately retrievable when accessed even if some segments are missing. Regenerating the missing erasure set segments is performed in a self-healing, cluster-initiated manner (similar to the recovery process for replicated objects) to protect against further disk loss. This process kicks off automatically when a missing volume is detected and automatically regenerates any missing segments.

Swarm always seeks the widest possible distribution of content (whether replicas or EC segments) *regardless of settings*, to maximize data protection. Swarm applies the following $2p+2$ protection levels as possible:

1. One segment per subcluster ... up to p segments per subcluster
2. One segment per node ... up to p segments per node
3. One segment per volume

How much disk space EC saves

The amount of disk space (or *footprint*) used for erasure-coded objects depends on the ratio of data to parity segments in the specified encoding.

Use the following formula to roughly calculate the disk space expected to be used by an EC object with one set of erasure segments:

(total segments ÷ data segments) × object size	= object footprint
$((k+p) \div k) \times \text{GB}$	= total GB

How footprint changes with different EC encoding (versus 3 reps)

- 1 GB object with 5:2 encoding: $((5 + 2) \div 5) \times 1 \text{ GB} = 1.4 \text{ GB}$ (vs. 3 GB for replication)
- 3 GB object with 5:2 encoding: $((5 + 2) \div 5) \times 3 \text{ GB} = 4.2 \text{ GB}$ (vs. 9 GB for replication)
- 3 GB object with 7:3 encoding: $((7 + 3) \div 7) \times 3 \text{ GB} = 4.3 \text{ GB}$ (vs. 9 GB for replication)

Note

Additional system metadata is written with each EC segment which adds about 16 bytes per segment.

See [Hardware Requirements for Storage](#) for how to size and optimize hardware for erasure coding.

See [Cluster Protection Planning](#) for how to determine the required number of nodes and optimize EC performance in a cluster.

- [Troubleshooting Erasure Coding](#)
- [Implementing EC Encoding Policy](#)
- [Methods Affected by Erasure Coding](#)
- [Conversion between Content Protection Types](#)

Defining Swarm Admins and Users

- [Granting Swarm Access](#)
 - [Disabling SNMP](#)
 - [Caution](#)
- [Encrypting Passwords](#)
- [Updating Passwords](#)
 - [Caution](#)
 - [Swarm has never booted](#)
 - [Updating SNMP passwords](#)
 - [Updating Swarm admin password](#)
 - [Changing admin password](#)

Granting Swarm Access

Swarm uses two pairs of security lists to grant access to storage cluster management and viewing:

- **Administrators** can access the Swarm UI and change the cluster configuration. SNMP *read/write* access is handled separately.
- **Operators** can view the Swarm UI. SNMP *read-only* access is handled separately.

i **Disabling SNMP**

Disable the Swarm Storage setting `snmp.enabled` if SNMP needs to be disabled cluster-wide, such as for a security need or using Swarm in containers. (v12.0)

Each user list is specified by a [configuration parameter](#) with name/value pairs in the Swarm Storage configuration file (`cluster.cfg` (CSN) or else `node.cfg`). Those passwords needed for SNMP access are handled in separate settings (v10.0):

```
security.administrators = { 'admin': 'adminpassword', 'admin2': 'adminpassword2' }
security.operators = { 'operator': 'operatorpassword', 'operator2': 'operatorpassword2' }
snmp.roCommunity = public
snmp.rwCommunity = ourpwdofchoicehere
```

or section notation:

```
[security]
administrators = { 'admin': 'adminpassword', 'admin2': 'adminpassword2' }
operators = { 'operator': 'operatorpassword', 'operator2': 'operatorpassword2' }
```

```
[snmp]
roCommunity = public
rwCommunity = ourpwdofchoicehere
```

Setting name	Default	Notes
--------------	---------	-------

security.administrators	<pre>{ 'admin': 'ourpwdofchoicehere' }</pre>	<p>One or more username:password pairs. Sets credentials for who can administer the cluster via the Swarm UI.</p> <p>Upgrading from 9.x – Remove the <code>snmp</code> username from here and update <code>snmp.rwCommunity</code> with the password if the value includes the <code>snmp</code> username.</p> <ul style="list-style-type: none"> • Example: <code>{ 'admin': 'adminpassword', 'admin2': 'adminpassword2' }</code>
security.operators	<pre>{ }</pre>	<p>One or more username:password pairs. Sets credentials for who can view the Swarm UI.</p> <p>Upgrading from 9.x – It is ignored if the value includes an <code>snmp</code> username; remove it from here and update <code>snmp.roCommunity</code> with the password.</p> <ul style="list-style-type: none"> • Example: <code>{ 'operator': 'operatorpassword', 'operator2': 'operatorpassword2' }</code>
snmp.rwCommunity	<pre>ourpwdofchoicehere</pre>	<p>String. Password for the SNMP read-write community.</p> <p>Required – The SNMP read-write password must be known to dynamically change the Swarm 'admin' password via SNMP. The config file <i>must</i> be edited to change the SNMP read-write password. The SNMP password is the sole option if the admin-level credentials are lost.</p>
snmp.roCommunity	<pre>public</pre>	<p>String. Password for the SNMP read-only community.</p>

Caution

- The name `admin` is reserved, so do not delete it, which can cause errors and affect performance. Define a complex password for protection if deciding not to use `admin`.
- Swarm prevents cluster booting if the SNMP security administrator (read/write user) is not set properly in the configuration file.
- All administrative users and passwords must agree on *all nodes* or certain cluster actions fail.
- Password updates are not complete until they are persisted in the cluster settings file across all nodes, and rapid, successive updates cannot be accepted on a given node until the first update completes processing.
- Change passwords from the **defaults** *before* putting the cluster in to production, and improve security by encrypting the Swarm passwords. *See next.*

Encrypting Passwords

Represent the password as a hexadecimal-encoded [MD5](#) hash of the following string instead of a clear text password:

```
username:user-list-name:password
```

Where username and password consist of ASCII characters and `user-list-name` can be either "CASstor administrator" or "CASstor operator".

To create the MD5 hash, use a programming language or a utility such as [md5sum](#) or Apache [htdigest](#). To update a node or cluster configuration file with a password hash created using `htdigest`:

1. Create a file containing a hash of the user name, password, and user list name:

```
htdigest -c password-file.txt "CAStor administrator" Jo.Jones
```

2. Enter and verify the user's password when prompted by **htdigest**.
3. Open the new file (`password-file.txt`) in a text editor. The hash is the *last* entry in the string:

```
Jo.Jones:CAStor administrator:08b0468c1d957b7bac24463dd2191a2d
```

Updating Passwords

The list of Administrators and passwords may be modified without rebooting by using several read-write SNMP OIDs. New administrative users can be added and existing users modified with the `addModifyAdministrator` SNMP OID. These are the essential commands:

- **Add admin users** – Include the new user name and password separated by a colon:
`addModifyAdministrator = "Jo.Jones:password1"`
- **Update password** for an existing user – Include the existing user name and new password separated by a colon:
`addModifyAdministrator = "Jo.Jones:password2"`
- **Delete admin users** (except the default admin and snmp users) – Send the name of an admin user:
`removeAdministrator = "Jo.Jones"`



Caution

- All administrative users and passwords must agree across *all nodes* or certain cluster actions fail.
- Any changes made via SNMP against a running cluster must be made in the node/cluster configuration file so any nodes offline when the change is made or new nodes added to the cluster after the fact can correctly authenticate cluster-wide actions.
- It can take several minutes for these SNMP changes to propagate in the cluster. During this update window, old passwords and deleted users continue to work for up to 10 minutes.

Important: How passwords are updated depends on which ones need updating and whether Swarm has ever been started.

Process	Examples and notes
---------	--------------------

<p>Swarm has never booted</p> <ol style="list-style-type: none"> 1. Create and hash an admin password. 2. Update passwords in the config file. 3. Important: Unmount/stop the USB drive or else the changes are not saved if booting from a USB flash drive. 4. Boot the Swarm cluster. 5. the password can be removed from the config file after the cluster is running. 	<p>Hash of password</p> <pre>\$ echo -n 'admin:CAStor administrator:NEWPASSWORD' md5sum cut -d ' ' -f1 7fe563b8532b3a460def0895895eebf5</pre> <p>The first time the cluster is booted the Swarm admin password <i>must</i> be in the config file:</p> <pre>[security] administrators = {'admin':'7fe563b8532b3a460def0895895eebf5'}</pre> <p>When the cluster is running, Swarm stores the admin password in the persisted Settings object, at which point it is safe to remove the password from the configuration file for security purposes:</p> <pre>[security] administrators = {}</pre>
<p>Updating SNMP passwords</p> <ol style="list-style-type: none"> 1. Update passwords in the config file. 2. Reboot the Swarm cluster. 	<p>Important – The SNMP read-write password must be known to dynamically change the Swarm 'admin' password. The config file <i>must</i> be edited if the SNMP read-write password needs to be changed..</p> <p>Proceed to change the Swarm 'admin' password after rebooting with the new SNMP password in the file</p>

<p>Updating Swarm admin password</p> <ol style="list-style-type: none"> 1. Create and hash an admin password. 2. Update password via SNMP, which Swarm saves in the persisted Settings object. 	<p>Changing admin password</p> <pre>snmpset -v2c -c SNMP- password -m +CARINGO-CASTOR-MIB SWARM-NODE-IP addModifyAdministrator s "admin:new- password"</pre> <pre>snmpset -v2c -c ourpwdofchoicewhere -m +CARINGO-CASTOR-MIB 172.20.3.85 addModifyAdministrator s "admin:7fe563b8532b3a460def0895895eebf5"</pre>
---	---

Frequently asked questions:

- *How do I change the active SNMP read-write password?* The SNMP passwords cannot be changed dynamically. Changing one or both requires a config file update and a cluster reboot.
- *What is the SNMP read-only password?* The read-only password 'public', which is the 'community string'
- *Is the read-only SNMP password in the persisted Settings object?* No
- *Can my SNMP read-write passwords in the persisted Settings object and cluster.cfg be different?* Yes, but the config file SNMP read-write password is used.
- *How do I change my admin password?* Update the password using SNMP and then update it in the config file unless it is removed from there.
- *How do I change my SNMP read-only password to the cluster?* Change the `snmp.roCommunity` setting in the config file and reboot the cluster.

Managing Volumes

This section describes how to manage the volumes in your storage cluster nodes.

Note
In normal operations, managing Swarm volumes does *not* require administrative actions. Special cases can occur if a volume or a node has a problem or if you decide to perform hardware maintenance on a node.

- [Moving Volumes between Nodes](#)
- [Replacing Failed Drives](#)
- [Retiring Hardware](#)
- [Retiring Volumes](#)
- [Returning a Stale Volume to Service](#)
- [How Swarm Responds to Disk Changes](#)

Moving Volumes between Nodes

Physical volumes can be moved between nodes as necessary to address hardware failures or other constraints as determined by an administrator.

The remaining cluster nodes immediately guarantee the correct number of replicas exist for all objects in the cluster when a volume goes off-line because of a volume failure, node failure, or node shutdown. The remaining nodes recognize the volume has returned to service and cease efforts to replicate the volume's data if a volume or node returns to the cluster during this procedure and prior to the 14-day time limit.

Important

Verify the node has enough RAM to handle volumes when adding or relocating volumes to a node. A node may be unable to mount some of the volumes if not.

Warning

Do not move Swarm disks between disk array controller types after they are formatted by Swarm. Each controller reports available disk space to Swarm that is matched with the controller. Many controllers claim the last section of the disk, reducing the total available space. A new controller may claim additional disk space not reported to Swarm if you switch your disks with another controller, so Swarm may attempt to write data to non-existing space, generating I/O errors.

Moving clusters

Volumes can also be moved to nodes in a different cluster. The objects on the volume become part of the new cluster and are checked for the correct constraints within the context of the new cluster when this occurs.

Replacing Failed Drives

Swarm volumes can be replaced after either an admin-initiated Retire (see [Managing Chassis and Disks](#)) or a Swarm-initiated failure resulting from I/O errors (see [Retiring Volumes](#)). A volume can be replaced after marked either **Retired** or **Unavailable**.

Administrators can insert a drive into a running node without restarting the server, provided the server hardware supports this function and `disk.volumes = all` is configured. This feature (called *hot plugging* or *hot swapping*) allows adding storage capacity to a node at any time.

See [Hot Swapping and Plugging Drives](#).

Identifying the Drive

The physical drive light turns on and stays on for one hour when a volume is marked unavailable or retired. Use the drive light features of the UI when identifying a failed or failing drive:

- **Swarm UI:** Click through **Cluster > Hardware** to view the chassis, and enable the drive light. Click the disk light toggle in the summary row to flash the drive light for a specific drive. Drive lights remain lit until turned off when enabling manually. See [Managing Chassis and Drives](#).
- **Legacy Admin Console:** The **Identify** feature allows identification of a **Retired** volume that need to be replaced. Use process of elimination if the volume was marked **Unavailable**: identify each of the working volumes in the chassis to determine which one does not flash and therefore needs to be replaced.

Remove the drive and verify the serial number with the message in the UI once the correct drive has been identified. Swarm recognizes a new volume is available and formats it for use when a new drive is inserted.

See [Drive Identification Plugin](#).

Suspending Volume Recovery

Suspend volume recovery while replacing a failed hard drive:

- **Swarm UI:** Administrators can suspend an in-process volume recovery using the **Suspend Recovery** option under the settings (gear) icon in the Swarm UI. Resume the recovery using either the **Enable Disk Recovery** button in the banner message or the **Enable Recovery** under the settings gear icon after the drive is replaced.

Tip

For drive-related events requiring user action (such as drive removal), Swarm helps locate the hardware by including the SCSI locator (bus ID) and volume serial number in the log message displayed in the UI. (v9.2)

- **Legacy Admin Console:**
 1. Select **Volume: Suspend Recovery** in the **Settings** menu.
 2. Remove the defective drive and install the replacement drive.
 3. Verify the new drive appears in the Swarm Admin Console and has a non-zero stream count after several minutes of cluster activity.
 4. Turn off **Volume: Suspend Recovery** in the **Settings** menu.

Retiring Hardware

- [Retire a Chassis](#)
- [Retire an Enclosure](#)

These are typical reasons to retire hardware:

- **Bad drives** – Retiring volumes that are throwing errors or whose performance has dropped. Such volumes should not be returned to service in the cluster. (See [Retiring Volumes.](#))
- **Encryption-at-rest** – Moving content off chassis so the volumes can be reformatted to support encryption-at-rest. (See [Configuring Encryption at Rest.](#))

Reformatting for encryption

Reformat and remount the volumes if retiring volumes to implement encryption at rest. Contact DataCore Support for a utility to streamline this process. (v10.1)

- **Planned EOL and upgrades** – Replacing old, serviceable hardware with new equipment (a hardware refresh). Planned end-of-life for enclosures involves decommissioning the enclosure by retiring all chassis and moving the reformatted drives back in to service.

Why not hotplug? Although Swarm does support moving drives within a cluster for quick data migration, moving an entire set of drives from a server chassis or rack enclosure temporarily risks data loss because the hardware receiving the drives may hold several of the replicas/segments of the same object. In addition, moving drives requires a level of hardware compatibility, and some hardware situations do not support such drive moves:

- Use of incompatible RAID cards/tagging between chassis (especially true for those who emulate JBOD with single disk RAID-0 definitions).
- Inability of the controller in the new chassis/enclosure to recognize or otherwise work with the drives being moved (such as drive firmware vs. controller firmware, even with a pure HBA/JBOD setup).
- Inability of either the old and/or new equipment to properly support [hot plug for drives.](#)

Best practice

The safest way to move all data being stored in an end-of-life chassis or enclosure is to retire the chassis and then format and reintroduce the drives.

Retire a Chassis

Swarm retires all drives within a chassis when retiring. Drives can be reformatted and returned to service if in good shape.

1. From the Storage UI, select **Cluster > Hardware**, open the **Chassis Details**, and select **Retire** from the action (gear) menu.



2. Wait for all volumes to reach a status of "retired".
3. Stop the storage processes from the system console on the physical chassis: **System Control > 3. Stop Storage Processes**
4. Format any disk that may be returned to service: **Disk Volumes > ALL**

5. Shut down the node: **System Control > 2. Shutdown System**
6. Transfer reformatted drives to other chassis as appropriate. (See [Hot Swapping and Plugging Drives.](#))

Retire an Enclosure

1. Add the new equipment to the cluster.
2. Retire *each chassis* in the old enclosure following the procedure above to reclaim any serviceable drives.
3. Power down and remove the enclosure when every chassis is retired (with drives reformatted for reuse) and shut down.
4. The drive returns to service storing Swarm data when formatted drives are inserted into a chassis.



Note

Contact DataCore Support to review the cluster layout and put together a plan for disk migration that works best if retiring is not feasible for a deployment.

Retiring Volumes

- [Tip](#)
- [Triggers for Retire](#)
 - [Note](#)
- [Canceling an Ongoing Retire](#)
 - [Canceling retire on a specific volume](#)
 - [Canceling retire on all volumes](#)

Due to current sophisticated disk storage devices and interfaces, the underlying disk system performs many error detection steps, bad sector re-mappings, and retry attempts. There is little chance a deterministic set of steps can be performed to work around the failure if a physical error propagates to the Swarm software level. Additionally, there is no guarantee the extent of the error can be isolated or the continued use of a failing device allows the node to continue operating normally with its peripheral storage devices.

Because of these inherent challenges, Swarm takes the conservative approach of *retiring* a volume as soon as it receives a configurable number of I/O errors. If the configured number of additional errors are received during the retire (**disk.ioErrorTolerance**), Swarm immediately marks the volume as **Unavailable** and kicks off both the failed volume recovery process (FVR) and the erasure-coding recovery process (ECR) to relocate all objects on the volume.



Tip

If Swarm retires a disk automatically because of I/O errors, check the diagnostic data collected in the logs. For the Swarm UI, see [Managing Chassis and Drives](#). (v11.1)

Triggers for Retire

Swarm changes a volume's state to Retiring when:

- **Retire** is clicked next to a volume on the node status page in the Swarm UI (or the legacy Admin Console).
- **Retire Chassis/Node** is clicked, which retires all volumes on the node at the same time.
- The number of I/O errors specified by **disk.ioErrorToRetire** occur in the time period specified by **disk.ioErrorWindow**.

A **Retiring** volume accepts no new or updated objects. A volume remains in the Retiring state until all objects stored on that volume (including replicas) are moved to other volumes in the cluster. The Retiring state persists even if the node is rebooted. The object count may increase.

The volume state is changed to **Retired** and Swarm does not use the volume anymore when all objects are moved. At that point, remove the volume for repair or discard it.



Note

If there are continued I/O errors that exceed the number specified by **disk.ioErrorTolerance** when the volume is in the **Retiring** state, the volume state is changed to **Unavailable**, regardless of whether Swarm has finished moving objects to other volumes.

Canceling an Ongoing Retire

An ongoing retire can be cancelled by using the `castorCancelVolumeRetire` SNMP action. It takes a string to name a specific volume, or all.

Canceling retire on a specific volume

```
snmpset -v2c -c ourpwdofchoicehere -m ./CARINGO-MIB.txt:./CARINGO-CASTOR-MIB.txt  
192.168.99.100 castorCancelVolumeRetire s "/dev/sda"
```

Canceling retire on all volumes

```
snmpset -v2c -c ourpwdofchoicehere -m ./CARINGO-MIB.txt:./CARINGO-CASTOR-MIB.txt  
192.168.99.100 castorCancelVolumeRetire s "all"
```

Returning a Stale Volume to Service

The storage cluster is designed to automatically adapt when a volume (hard drive) or node fails for any reason. Swarm checks every storage cluster volume during the node startup procedure, and it tracks any gaps in service that trigger a status change:

- A **volume** is considered "stale" and the contents cannot be used unless an administrator overrides this process if disconnected from the cluster for more than 2 weeks.
- A **node** and all volumes are considered stale and cannot be used if it is shut down for more than 2 weeks.

The "stale" status is triggered by a service gap of 2 weeks, which is the default value for the [disk.obsoleteTimeout](#) setting.

Force a volume remount by modifying the [disk.volumes](#) setting and adding the :k (keep) policy option. Return them to service dynamically (either remounting or reformatting) using SNMP. (v9.3)

Reformatting volumes (recommended)

Reformatting the volume allows it to be filled by the health processor (HP) in an orderly fashion. Performing this prevents creating excessive work for the health processor and prevents generating trapped space needed to be reclaimed.

```
snmpset -v2c -c ourpwdofchoicehere -m ./CARINGO-MIB.txt:./CARINGO-CASTOR-MIB.txt
192.168.99.100 castorFormatStaleVolumeAction s "/dev/sda"
```

Important

The volume's [encryption status](#) is always retained on return to service; physical removal from Swarm is required to change it.

Remounting volumes

It is rarely desirable to remount a volume that has stale content. The volume's missing content is recovered by this time and the cluster has its full complement of replicas of the cluster's content. Adding extra replicas creates work for the health processor to sift through the replicas, cleaning up redundant and obsolete copies. This cleanup creates trapped space in the cluster that take several HP cycles to reclaim.

Note

Content explicitly deleted by clients can be inadvertently resurrected when forcing a stale volume back in to service. This is not a problem for content automatically deleted by lifepoint policies because the obsolete content is discovered and deleted by the Swarm health processor.

```
snmpset -v2c -c ourpwdofchoicehere -m ./CARINGO-MIB.txt:./CARINGO-CASTOR-MIB.txt
192.168.99.100 castorRemountStaleVolumeAction s "/dev/sda"
```

How Swarm Responds to Disk Changes

Note

Swarm requires control of all volumes (`disk.volumes = all`) to support hot plugging.

- I/O errors are recorded in the log if the Health Processor is actively scanning a disk when it is removed. These errors are expected and do not indicate a problem.
- When a disk is removed, volume recovery (FVR) and erasure coding recovery (ECR) are both triggered, which includes creating new replicas or erasure set segments for objects stored on that disk.
- Both recovery processes stop if a disk is inserted in the same node or in a different node in the cluster. There is a temporary state of over-replication because the returned volume has replicas or segments already recreated elsewhere. In time, the excess replicas or segments are deleted.
- Swarm recognizes, formats, and mounts a disk as a new volume if a non-Swarm disk is inserted in a node.
- A Swarm-formatted disk continues to function as a volume without loss of data if inserted, either into the same node or into a different node.
- A volume remains retired if a previously retired Swarm-formatted disk is inserted. No manual configuration or intervention is required.
- Messages display in logs and in the Swarm Admin Console to indicate a disk was inserted or removed.
- Wait 2 minutes between disk insertions to guarantee new disks are evenly distributed across multiple nodes running on the chassis if inserting multiple disks into the same server chassis.

Caution

When adding or relocating volumes to a node, verify the node has enough RAM to handle them, else the node may be unable to mount some of the volumes.

Warning

Do not move Swarm disks between disk array controller types after they are formatted by Swarm. Each controller reports available disk space to Swarm matched with the controller. For example, many controllers claim the last section of the disk, reducing the total available space. The new controller may claim additional disk space not reported to Swarm, so Swarm may attempt to write data to non-existing space, generating I/O errors if disks are switch with another controller.

Using SNMP with Swarm

The Swarm SNMP agent implementation allows you to monitor the health of cluster nodes, collect usage data, and control node actions. You can integrate your storage cluster into an enterprise SNMP monitoring infrastructure. Swarm supports SNMP version 2 only.

Disabling SNMP

If you need to disable SNMP cluster-wide, such as for a security need or using Swarm in containers, disable the Swarm Storage setting `snmp.enabled`. (v12.0)

Persisted settings

Many configuration parameters are persisted and are set on a running cluster using the Swarm UI or SNMP. See [Persisted Settings \(SNMP\)](#).

SNMP MIBs

If you boot from a Platform Server, see the following MIBs located at `/usr/share/snmp/mibs`:

- **CASTOR-MGR-MIB.txt**. An aggregate MIB for all cluster nodes.
- **CARINGO-CASTOR-MIB.txt**. A standard Swarm hardware MIB provided with the Swarm SNMP agent

If you do not boot from a Platform Server, see the **CARINGO-CASTOR-MIB.txt** MIB located in the root directory of the Swarm software distribution.

Swarm allows you to access the standard hardware MIBs distributed with the Net-SNMP package. These MIBs provide hardware reporting for areas such as processor load, memory availability, and network bandwidth.

For details on the available OIDs, see the [Net-SNMP MIB documentation](#).

SNMP Tools and Monitoring Systems

- [SNMP version](#)
- [Open Source Tools](#)
- [SNMP Examples with Swarm](#)
 - [Change in snmpwalk](#)
- [SNMP Action OIDs](#)
 - [Important](#)
- [Practical SNMP with Swarm](#)
 - [Tip](#)
 - [Health Monitoring](#)
 - [Capacity Monitoring](#)
 - [Tip](#)
 - [Client Activity Reporting](#)
- [SNMP Repository Dump](#)
 - [Accessing the Repository Dump](#)
 - [Disk Monitoring](#)
 - [Discontinued Items](#)



SNMP version

Swarm supports SNMP version 2.

Any standard SNMP query tool and monitoring system can be used to interact with Swarm. The examples in this section use the open source Net-SNMP (formerly UCD-SNMP) package available for UNIX and Microsoft Windows platforms. Install the Swarm MIB definition file before using most tools and monitoring packages. Follow the instructions included with the tool or package for more information.

Open Source Tools

The following tools can be useful to monitor and manage Swarm. DataCore does not endorse the applicability nor the fitness of these products when used within any environment.

- **Net-SNMP** (net-snmp.sourceforge.net). Provides command-line tools for UNIX and Windows environments to send and receive SNMP requests.
- **Nagios** (nagios.org). Provides web-based monitoring system for UNIX environments for monitoring systems and sending alerts through email and pager.
- **Zenoss** (zenoss.com). An SNMP-based system for IT monitoring and management.

SNMP Examples with Swarm

Complete the following to prepare to use the examples in this section:

1. Record the IP address of a storage cluster node. Record the SCSP Proxy if the cluster is not in the subnet. The node's IP address is `172.16.0.32` in the examples below.
2. Run the command from the directory containing **CARINGO-CASTOR-MIB.txt**.
Copy **CARINGO-CASTOR-MIB.txt** from the root directory of the USB flash drive or distribution to a local directory.
3. Record the following passwords:
 - *read-only-password*. The password for the read-only user defined in the **security.operators** [setting](#). Default: `public`
 - *read-write-password*. The password for the read-write user defined in the **security.administrators** [setting](#). Default: `ourpwdofchoicehere`

See [Defining Swarm Admins and Users](#).



Change in snmpwalk

The 7.2 release changed the snmpwalk of the whole CASTOR MIB to make it skip several large, detailed tables in SNMP groups to protect cluster performance. Administrators must upgrade from CSN v6.5 to update the CSN reporter.

Create a targeted snmpwalk request if data from those skipped tables is needed. The **snmp.getnextskips** setting directs top-level snmpwalk to skip the groups and tables under the following: `clusterConfig`, `responseHistogramTable`, `hp`, `clusterdata`, `indexer`, `configVariableTable`, `castorFeeds`, `feedVolTable`, `performance`, `recoveryTable`

SNMP walk (snmpwalk) of all Swarm values on a node:

```
snmpwalk -v 2c -c read-only-password -m +./CARINGO-CASTOR-MIB.txt 172.16.0.32 caringo
```

Request for a specific SNMP variable from a Swarm node:

```
snmpget -v 2c -c read-only-password -m +./CARINGO-CASTOR-MIB.txt 172.16.0.32 reads
```

Set request to shut down a Swarm node:

```
snmpset -v 2c -c read-write-password -m +./CARINGO-CASTOR-MIB.txt 172.16.0.32
castorShutdownAction s shutdown CARINGO-CASTOR-MIB::castorShutdownAction = STRING: "shutdown"
```

Set request to change the cluster's `sleepAfter` setting to 7260 seconds (121 minutes):

```
snmpset -v2c -c read-write-password -m +./CARINGO-CASTOR-MIB.txt 172.16.0.32
sleepAfter i 7260
```

SNMP Action OIDs

The "action" OIDs in Swarm are the SNMP objects affecting the operation of a node or the cluster.

Important

The action is recommended to be written to a single node to allow updates to the persisted settings UUID from a single node to prevent conflicts for cluster-level parameters such as **volumeRecoverySuspend**.

castorFeedRestartAction	Restarts a feed on a node using SNMP. The feed restarts on all nodes in the cluster when setting the OID value to a specific feed value. The castorFeedTable OID allows viewing the Swarm feed information for a specific node. Each entry indicates a feed running on the selected node. The Admin Console allows viewing the SNMP Repository Dump page, which provides node-specific information.
logHost	Sets the logging host for writing log messages. A node sets the logging host based on the loghost parameter when booted. Redirect syslog messages to a workstation to debug an issue.
logLevel	Sets the logging level. A node sets the logging level based on the loglevel parameter when booted. Increase the logging level to debug an issue and then return the level to the previous value when completed.
nodeLogLevel	Sets the logging level for a specific node in the cluster, overriding the boot configuration specified by the loglevel parameter as well as the cluster-wide logLevel object.
logForceAudit	Sets forced audit logging for all nodes in the cluster, independent of the overall log level.
castorRetireAction	Removes the contents of a disk volume or an entire node in an orderly fashion. Consider retiring disks to save content not saved on another disk instead of removing disks. The device name from the node configuration vols parameter or the all string is written to this OID. Volumes from multiple nodes in the cluster can be simultaneously retired.
castorShutdownAction	Sets a graceful shutdown or reboot a node or an entire cluster. The supported values are: <ul style="list-style-type: none"> • shutdown. Shuts down this node. • reboot. Reboots this node. • clustershutdown. Shuts down all nodes in the cluster. • clusterreboot. Reboots all nodes in the cluster.
volumeRecoverySuspend	Suspends volume recovery and erasure coding recovery behavior in the cluster during an upgrade or a network outage.

Practical SNMP with Swarm

This section outlines practical approaches in using the built-in SNMP agent to monitor the health and operational aspects of a storage cluster.

Tip

Although an ICMP ping monitor of a Swarm node can be set up, using the SNMP variables provides detailed indications of disk and capacity problems.

Health Monitoring

The following variables can be used to monitor the basic health of a Swarm node. The volume table has n from 1 to the number of volumes.

- **caringo.castor.castorState.** Equal "OK."
- **caringo.castor.castorVolTable.volEntry.volState.n.** Equal "OK."
- **caringo.castor.castorVolTable.volEntry.volErrors.n.** Equal to zero.

There is something wrong with the node if the monitoring console receives timeouts when trying to read these variables. The node or the disks are transitioning from the normal state if the state values are anything other than "ok".

	Node	Volume
Valid states	OK	OK
	Retiring	Retiring
	Retired	Retired
		Unavailable

Any non-zero value in the volume error count indicates a hard error has surfaced from the hardware through the OS driver and to the Swarm process.

Capacity Monitoring

The following variables can be monitored and collected for capacity alerting and reporting. The volume table has n from 1 to the number of volumes.

- **caringo.castor.castorFreeSlots.** Greater than zero.
- **caringo.castor.castorVolTable.volEntry.volMaxMbytes.n**
- **caringo.castor.castorVolTable.volEntry.volFreeMbytes.n**
- **caringo.castor.castorVolTable.volEntry.volTrappedMbytes.n**

The **castorFreeSlots** variable indicates how many more objects a node can hold before it exhausts the memory index. The node is unable to store additional objects until objects are deleted or moved to other cluster nodes (or more RAM is added to the node) if this occurs. The free slots indicate how much RAM is required per object.

See the [Memory Sizing Requirements](#) for RAM effects on node storage.

Add the values **volFreeMbytes** and **volTrappedMbytes** to compute the amount of disk space available for writing content.

$$(\text{volFreeMbytes} + \text{volTrappedMbytes}) / \text{volMaxMbytes} = \% \text{ free space on a disk volume}$$

$$\text{volUsedMbytes} / \text{volMaxMbytes} = \% \text{ space used by current context}$$

Tip

Total these disk usage variables for all volumes in a node and all nodes in a cluster to produce capacity utilization reports.

Client Activity Reporting

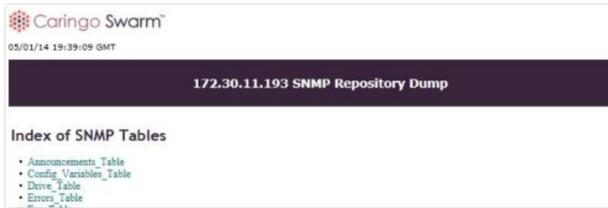
Collect and report the amount of client activity received by the nodes to understand the end-user usage patterns and identify nodes receiving significantly more activity than others. The resulting value can indicate a poor primary access node selection mechanism in the client application code.

The following SNMP variables indicate client request activity on a Swarm node.

- caringo.castor.scsp.writes
- caringo.castor.scsp.reads
- caringo.castor.scsp.infos
- caringo.castor.scsp.deletes
- caringo.castor.scsp.errors
- caringo.castor.scsp.updates
- caringo.castor.scsp.copies
- caringo.castor.scsp.appendds

SNMP Repository Dump

The SNMP Repository Dump page provides additional node-specific information.



Accessing the Repository Dump

Access the SNMP Repository Dump page for a cluster node:

1. Open the legacy Admin Console.
2. In the **Node IP** column, click the IP address of the target node.
3. Scroll down and maximize **Node Info**.

4. Scroll down and click **SNMP Repository**.

See the SNMP MIB Reference file included in the Swarm download bundle for more on the SNMP Repository Dump tables.

Disk Monitoring

Swarm 12 collects more health data from the SMART values reported by storage disks, can be accessed via the SNMP Drive table. (v12.0)

- **driveStatus** is now correctly computed.
- **drivePowerOnHours** is from SMART attribute 9.
- **driveTempC** is from SMART attribute 194.
- **driveCompromisedCount** is the sum of SMART attributes 5, 187, 188, 197, and 198. A non-zero value may indicate an impending disk failure.

Discontinued Items

Note these SNMP items are no longer populated (v9.4):

- planarTemp
- tempStatus
- fanRedundancy
- psuRedundancy
- instantaneousWatts
- instantaneousMA
- minPowerCap
- maxPowerCap
- nics
- nicTable (including detail)
- nicFWVsn
- driveTable.driveStatus
- fans
- fanTable (including detail)
- psus
- psuTable (including detail)
- powerIntervals

- powerDrawTable (including detail)

Those SNMP values can be re-populated with a configuration change if relying on them. Contact DataCore Support for instructions.

Managing Swarm Nodes

- [SNMP Commands](#)
- [Shutdown Action for Nodes](#)
 - [Tip](#)
 - [Note](#)
- [Retire Action for Nodes and Volumes](#)
 - [Note](#)
 - [Single volumes](#)
 - [Entire node](#)
 - [Warning](#)

SNMP Commands

Storage cluster nodes are controlled through the SNMP action commands. The following OIDs allow disabling nodes and volumes with nodes from a storage cluster:

- **castorShutdownAction**. Disable nodes and volumes within nodes for servicing.
- **castorRetireAction**. Disable nodes and volumes within nodes for retirement.

Shutdown Action for Nodes

To gracefully shut down a Swarm node, the string `shutdown` is written to the **castorShutdownAction** OID. Writing the string `reboot` to this OID causes a Swarm node to reboot.

A node initiates a graceful stop by unmounting all volumes and removing itself from the cluster when it receives a shutdown or reboot action. The node is powered off if the hardware supports this action for a shutdown. The node reboots, re-reads the node or cluster configuration files, and starts up Swarm for a reboot.

A graceful shutdown is required to perform a quick reboot. Performing an ungraceful shutdown forces the node to perform consistency checks on all volumes before rejoining the cluster.

Tip

Before shutting down or rebooting a node, check the node status page or the SNMP **castorErrTable** OID for critical error messages. Any logged critical messages are cleared upon reboot.

Note

Wait at least 10 seconds in between each node reboot if rebooting more than one node at a time but not the whole cluster. This pause verifies each node can communicate the rebooting state to the rest of the cluster, so other nodes do not initiate recovery for the rebooting node.

Retire Action for Nodes and Volumes

The Retire action is used to permanently remove a node or a volume within a node from the cluster. This action is intended for retiring legacy hardware or preemptively pushing content away from a volume with a history of I/O errors. Retired volumes and nodes are visible in the Swarm Admin Console until the cluster is rebooted.

See [Retiring Volumes](#).



Note

The Retire action may take an extended amount of time to complete and requires at least three health processor cycles.

Single volumes

All stored objects are moved to other nodes in the storage cluster when a volume is retired. The volume becomes a read-only volume and no additional objects can be stored on it after initiating a volume retirement. The volume is idled with no further read/write requests after all objects are moved to other locations in the cluster.

Each volume is given a unique name within the node – the device string from the `vols` line in the configuration file. To retire a volume, the name is written as a string to the `castorRetireAction` OID. The volume retirement process is initiated immediately upon receipt and the action cannot be aborted after it starts.

To manually retire a volume,

1. Open the [Swarm UI](#) (or legacy Admin Console).
2. Click the targeted chassis/node (IP address).
3. For the targeted disk/volume, select **Retire**.

Entire node

Retiring a node means all volumes on the node are retired at the same time. After all volumes in the node are retired and the node data is copied elsewhere in the cluster, the node is permanently out of service and does not respond to further requests.

To retire a node and all volumes, the `all` string is written to the `castorRetireAction` OID. The node retirement process is initiated immediately upon receipt and the action cannot be aborted after it starts.



Warning

Verify the cluster has enough free space *and* nodes to store the objects from the retiring volume. For subclusters, this applies to the subcluster where the retiring volume resides. The retiring node cannot complete the retirement process until adding additional nodes if the number of nodes in the cluster or subcluster do not have enough space to store at least two replicas of all objects. The Retire action does not require the configured default replicas (`policy.replicas default`) are maintained to complete retirement. Messages are logged indicating sufficient replicas cannot be created if there are not enough nodes to maintain the minimum number of replicas.

Statistics for Logical Usage

- [Tip](#)
- [Usage via SNMP and REST](#)
 - [Note](#)
 - [Note](#)
 - [SNMP for usage](#)
 - [REST call for usage](#)
- [Usage via Metrics](#)

Swarm calculates storage use by addressable object to support conventional (storage filer) reporting of file counts and space usage. This approach tracks cluster-wide capacity by counting **logical objects** (the unique content of uploaded and versioned files) rather than actual **streams** (the raw space that is consumed by *all* Swarm components, including replicas, EC segments, context objects, and manifests). The versions add to the object totals; if a cluster held one 20 MB image with 3 replicas and 4 prior versions: 5 logical objects and ~100 MB logical space is consumed.

Swarm continuously sends the nodes updates about the cluster's logical usage (the current number of objects and the space they consume), which the nodes update with the local space-affecting activity. Swarm aggregates these updates (for accuracy) and publishes them using SNMP and REST as `logicalObjects` and `logicalSpace`. A third statistic, `logicalUnprocessed`, exists to provide insight in to the accuracy of the other statistics (the closer to zero, the more accurate). Swarm propagates this data quickly, so there is little lag behind the cluster activity affecting usage: writes, deletes, and updates. A drop in aggregated estimates is reflected after a disk failure, followed by an increase to the true value, once Volume Recovery recreates the lost streams previously on the disk.

Tip

When first booting the cluster after installing or upgrading to version 9.0, Swarm starts traversing the volumes to build these statistics, so they are not accurate until that completes; the value of `logicalUnprocessed` indicates the progression. Expect it to take 1 complete HP cycle to drop `logicalUnprocessed` to 0.

Usage via SNMP and REST

Swarm aggregates usage statistics from each volume and publishes them as cluster-wide values:

Aggregates	Units	Description	Accuracy
logicalObjects	count	The number of unique objects (including historical versions) stored for the entire cluster. Each content object counts only as 1, regardless of the number of replicas or EC segments that comprise it.	Approaches the actual number of logical objects in the cluster, minus context (domain, bucket) objects. <div data-bbox="884 1570 1453 1843" data-label="Text"> <p>Note</p> <p><i>Logical counts are estimates, and they are not accurate during volume recovery.</i> The estimating is a consequence of Swarm's robust, no-single-point-of-failure design: Swarm keeps no master list of objects, so counts are inferred from multiple overlapping sources of information.</p> </div>

logicalSpace	MB	The logical space stored for the entire cluster, including historical versions (which are separate objects).	Includes both the data and the persisted headers on each object, with header newlines counting as two characters ('\r\n'). EC encoded objects may include a small overage.
logicalUnprocessed	count	The number of streams in the cluster <i>not</i> accounted for in <code>logicalObjects</code> and <code>logicalSpace</code> . After implementation, it drops until it catches up, approaching zero.	When compared to the number of streams in the cluster, allows rough verification of other statistics, especially following the first boot after it is implemented.

Note

These are cluster-level statistics, so each node is publishing the same values.

Get `logicalObjects`, `logicalSpace`, and `logicalUnprocessed` by polling a node using SNMP:

SNMP for usage

```
snmpget -m +CARINGO-CASTOR-MIB -v2c -M +/usr/share/snmp/mib2c-data -cPASSWORD -OQs {node- ip} logicalSpace
```

Get `logicalObjects`, `logicalSpace`, and `logicalUnprocessed` by polling a node using the REST API:

REST call for usage

```
http://{node- ip}:91/api/storage/clusters/{clustername}
```

Trends – Each volume in a Swarm cluster is computing partial statistics for logical objects with replicas on other volumes. Swarm works to keep the correct number of replicas (and EC segments) for every object, but, if there are too many replicas, the statistics trend higher. In the case of hardware failure, the statistics trend lower while the recovery is taking place.

Timing – Each volume has accurate partial statistics immediately after a write or delete. REST API statistics are immediately available after each volume broadcasts messages that are sent every 30 seconds, but SNMP adds up to another 60 seconds for periodic polling of the aggregated values. Metrics does not aggregate, so the periodic metrics reports is current with respect to the accounting cursor.

Usage via Metrics

Usage statistics are reported using Swarm's [Metrics](#) mechanism. These metrics are checked on demand, at query time. Although Swarm publishes the statistics under the **volume** metrics, the values represent the cluster level:

Volume Metrics	Units	Description
logical_objects	count	The number of unique objects (including historical versions) stored for the entire cluster. Each content object counts only as 1, regardless of the number of replicas or EC segments that comprise it.

logical_space	bytes	The logical space stored for the entire cluster. The <code>logical_space</code> value is in bytes, not MB, for greater accuracy.
logical_unprocessed	count	The number of streams (replicas, EC segments, etc.) in the cluster <i>not</i> counted for <code>logicalObjects</code> and <code>logicalSpace</code> .

Troubleshooting Storage

This section provides information to help you troubleshoot and resolve issues in your Swarm storage cluster.

- [Operational Problems](#)
- [Troubleshooting Boot Errors](#)
- [Troubleshooting Configuration](#)

Operational Problems

For disk-related events requiring user action (such as disk removal), Swarm helps locate the hardware by logging the SCSI locator (bus ID) and volume serial number at CRITICAL and ANNOUNCE log levels, which makes them display in the UI. (v9.2)

Helpful statistics

Swarm keeps statistics on incomplete read and write requests, which can help diagnose clients behaving incorrectly.

- **SNMP:** `clientPrematureCloseRead`, `clientPrematureCloseWrite`
- **UI:** Drill in to the health reports for chassis-level statistics. For the legacy Admin Console, statistics appear on a node's status page, under **Node Operations:** SCSP: Client premature close (read), SCSP: Client premature close (write)

Tip

For disk-related events requiring user action (such as disk removal), Swarm helps locate the hardware by including the SCSI locator (bus ID) and volume serial number in the log message displaying in the UI. (v9.2)

Symptom	Action
A volume device failed.	<p>Allow the node to continue running in a degraded state (lowered storage) OR Replace the volume at the earliest convenience.</p> <p>See Replacing Failed Drives.</p>
A node failed.	<p>Repair the hardware and return it to service within 14 days if a node fails but the volume storage devices are functioning properly.</p> <p>All volumes are considered stale and cannot be used if a node is down for more than 14 days. Force a volume to be remounted by modifying the volume specification and adding the <code>:k (keep)</code> policy option after 14 days.</p> <p>See Managing Volumes.</p>
<p>In the UI, all remaining cluster nodes are consistently or intermittently offline.</p> <p>Viewing the legacy Admin Console from different nodes, other nodes appear offline and unreachable.</p>	<p>Check the Swarm network configuration setting in each node (particularly the <code>group</code> parameter) to verify all nodes are configured as part of the same cluster and connected to the same subnet if a new node cannot see the remaining nodes in the cluster.</p> <p>Verify IGMP Snooping is enabled on the network switch if the network configuration appears to be correct. An IGMP querier <i>must</i> be enabled in the same network (broadcast domain) if enabled. In multicast networks, this is normally enabled on the router leading to the storage cluster, which is usually the default gateway for the nodes.</p> <p>See IGMP Snooping.</p>

<p>Read-only access to the UIs even though the user is listed in <code>security.administrators</code>.</p> <p>Cannot view the Swarm UI.</p>	<p>Added an operator (a read-only user) to <code>security.operators</code> but did not add the administrator user name and password to <code>security.operators</code> as well. The Swarm UI cannot be accessed as an administrator.</p> <p>Add all administrator users to the <code>security.operators</code> parameter in the node or cluster configuration file to resolve this issue.</p> <p>See Defining Swarm Admins and Users.</p>
<p>The network does not connect to a node configured with multiple NIC ports.</p>	<p>Verify the network cable is plugged into the correct NIC. Depending on the bus order and the order the kernel drivers are loaded, the network ports may not match the external labeling.</p>
<p>A node automatically reboots.</p>	<p>This issue may indicate a software problem if the node is plugged into a reliable power outlet and the hardware is functioning properly.</p> <p>The Swarm system includes a built-in fail safe that reboots itself if something goes wrong. Contact DataCore Support for guidance.</p>
<p>A node is unresponsive to network requests.</p>	<p>Perform the following steps until the node responds to network requests.</p> <ul style="list-style-type: none"> • Verify the client network settings are correct. • Ping the node. • Open the legacy Admin Console on the node by entering the IP address in a browser window (<code>http://{ip-address}:90</code>). • Attach a keyboard to the failed node and press Ctrl-Alt-Delete to force a graceful shutdown. • Press the hardware reset button on the node or power cycle the node.
<p>The cluster is using more data than expected.</p>	<p>Using Elasticsearch, enumerate the <code>CAStor-Application</code> field to determine how much data is being written by which application. Many Swarm applications use this metadata header, and having it indexed allows analyzing which application created which content.</p>
<p>A node is not performing as expected.</p>	<p>In the <code>castor.log</code>, view the node statistics, which include periodic logging of CPU utilization for each process:</p> <pre>2015-11-05 16:13:22, 898 NODE INFO: system utilization stats: pid_cpusys: 0.06, pid_cputot: 1.67, pid_cpuusr: 1.61, sys_contexts_rate: 5728.00, sys_cpubusy: 0.91, sys_cpubusy0: 0.37, sys_cpubusy1: 1.46, sys_cpuio: 0.02, sys_cpuirq: 0.01, sys_cpusys: 0.06, sys_cpuusr: 0.82</pre>

Troubleshooting Boot Errors

Symptom	Action
<p>When booting, the node generates an error stating a boot device is unavailable.</p> <p>The node boots into an operating system other than Swarm.</p>	<p>Verify the node is capable of booting from a USB device and the USB memory device is configured as the primary boot device if booting from a USB device.</p> <p>Verify the following if PXE booting:</p> <ul style="list-style-type: none"> • The server is configured to network boot. • PortFast is configured on the switch ports leading to the Swarm node. <p>The extended time delay required for listening and learning Spanning Tree states can prevent netboot from delivering the Swarm image to the node in a timely manner if the above are not satisfied.</p>
<p>The node boots from the USB device, but Swarm fails to start.</p> <p>The node begins to boot but reports a "kernel panic" error and stops.</p>	<p>These symptoms usually indicate a hardware compatibility issue with the hardware. Contact DataCore Support with the details of the hardware setup.</p>

Troubleshooting Configuration

Symptom	Action
<p>After the system boots, a message appears stating that the configuration file is missing.</p> <p>The node boots, but storage is not available on the node.</p> <p>A hard drive in a node does not appear as available storage.</p> <p>After adding a new hard drive to a node, some of the volumes do not mount.</p> <p>After moving a volume between nodes, some volumes in the new node do not mount.</p>	<p>Verify each node has a <code>node.cfg</code> file on the USB stick and that <code>disk.volumes</code> is properly specified.</p> <p>See Configuring the Nodes.</p> <p>The volume must be greater than the minimum value specified by the <code>disk.minGB</code> parameter (64 GB by default) or it does not boot. Small disks can be booted by lowering the size value in <code>disk.minGB</code>.</p> <p>If the <code>vols</code> specification is correct and the volume is greater than <code>disk.minGB</code>, this may be an issue with the amount of RAM in the node. Check the available RAM and verify it is sufficiently provisioned.</p>
<p>The node boots from the USB device but Swarm fails to start.</p> <p>The node begins to boot but reports a "kernel panic" error and stops.</p>	<p>These symptoms usually indicate a compatibility issue with the hardware. Contact Swarm Support with the details of your hardware setup.</p>
<p>Some changes to the <code>node.cfg</code> file disappear after editing.</p>	<p>If a USB flash drive is removed from a computer without unmounting, changes can be lost. Use the method for your OS to stop and unmount the USB media before removing it.</p>
<p>The following alert displays in the Swarm Admin Console:</p> <pre>Local clock is out of sync with node ip-address</pre> <p>The following indicator displays for each node where the error occurs:</p> 	<p>A clock synchronization issue exists between the cluster nodes.</p> <p>The clock icon displays next to any node that sends a data packet that is more than three minutes offset from the reporting node's local clock. The local node logs a critical error.</p> <p>Verify the Network Time Protocol (NTP) settings are correct and the cluster nodes can access the configured NTP server.</p> <p>See Configuring an External Time Server.</p>
<p>A node hangs during boot while initializing ACPI services.</p>	<p>System hardware is conflicting with the Advanced Configuration and Power Interface (ACPI) interface.</p> <p>To resolve this issue, add the argument <code>acpi=off</code> to the <code>syslinux.cfg</code> file on the USB flash drive (for local booting) or to the PXE configuration file (for network booting).</p>
<p>The Swarm node boots as having an unregistered license.</p>	<p>The license file is not in the Caringo directory on the USB drive, or the <code>licenseFileURL</code> option in the node or cluster configuration file is not set properly.</p>
<p>Inconsistent performance when using virtualized disks.</p>	<p>There may be timeout issues. Add the <code>disk.deviceTimeout</code> configuration parameter and increase the value as needed for your virtualization environment.</p>

Swarm Cluster Services (SCS)

Swarm Cluster Services (SCS) Server – now in its third generation – provides a single point to administer and control a Swarm cluster and its network.

- [SCS Overview](#)

SCS Overview

The Swarm Cluster Services (SCS) unifies and centralizes the services and management information needed to install, upgrade, and monitor Swarm storage clusters. Installed on a dedicated node, SCS simplifies the network services and DataCore product installation process for Swarm administrators.

The SCS infrastructure provides these essential features:

- **Required network services.** Provides network services that are configured to support a Swarm cluster: DHCP, PXE/Network Boot, TFTP, Syslog, and NTP.
- **Swarm configuration management.** Provides the ability to configure and update the cluster and individual chassis as needed.
- **Command-line interface.** The CLI provides a direct interface to SCS management tasks.

Limitations

There are a few functional limitations of SCS:

- Control of the Swarm SCS Server is limited to CLI access only. No UI support.
- No automated upgrade process from a CSN installation to a SCS installation.

Use Cases

SCS may be used in environments that meet the following requirements:

- Are managed by administrators familiar with running CSN and Swarm
- Have a staging environment with bare-metal storage nodes
- Have Swarm 14 installed

SCS Concepts

The SCS server is responsible in part for coordinating and maintaining configuration across the various elements of a Swarm site ecosystem. It is not oriented to any part of the ecosystem, and has a separate conceptual approach to managing the various elements.

- [Services Node for Swarm](#)
- [Swarm Orchestration](#)
- [Swarm Environment and Networking](#)
- [Network Architecture](#)
- [Swarm Interfaces](#)
- [Components](#)
- [Groups](#)
- [Instances](#)
- [Settings](#)
- [Templates](#)

Services Node for Swarm

The SCS server is an extensible services node for managing Swarm. On a single server, it configures the environment and coordinates the support services needed to deploy hardware into and support a Swarm cluster. It replaces the original Caringo Services Node (CSN).

An SCS server automatically installs and configures the network infrastructure required for the Swarm environment when brought online.

Swarm Orchestration

SCS orchestration spans initial Swarm setup and deployment as well as all ongoing maintenance.

- Installation and configuration of Swarm Storage software
- Network boot support
- Automatic provisioning of network and node configs

Swarm Environment and Networking

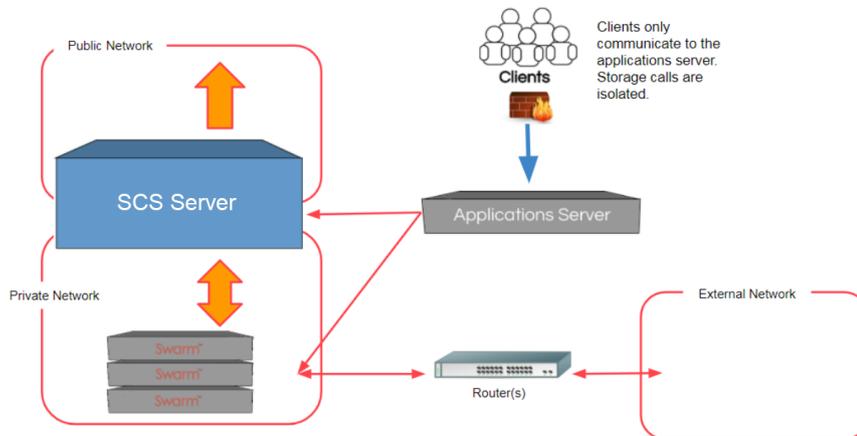
The following system services are set up and initialized by the initial configuration of the SCS server:

1. DHCP server, to allocate Swarm node addresses
2. TFTP server, for Swarm network boot
3. NTP time server, essential for Swarm clock synchronization (defaults to pool.ntp.org)
4. Syslog server, for centralized logging (log files can be found within: `/var/log/datacore`)
5. Configuration and License server, for Swarm node access

Network Architecture

Swarm architecture uses a private network dedicated to the Swarm Storage nodes and application servers. Access to this cluster is restricted to the SCS server and any applications residing on private subnet (in the absence of existing routers). SCS server owns this private storage network, but it can support flexible network topologies.

- SCS server addresses are IPs on a routable subnet.
- Application servers can be dual-homed like the SCS server, for ease of access to cluster, but the preferred architecture is to access the cluster via the gateway included in the SCS server.
- Swarm Storage nodes' default gateway is the SCS server, which can be changed to support existing routers. Direct routed access to storage nodes bypasses access control and is recommended to be avoided in most cases.



Swarm Interfaces

The SCS CLI is the sole interface to SCS functionality:

- **CLI** (Command-Line Interface) – see [SCS CLI Commands](#)
 - The CLI, called `scsctl`, is a native application installed on the SCS server.
 - To get CLI help, type: `scsctl help`

The SCS server is responsible in part for coordinating and maintaining configuration across the various elements of a Swarm site ecosystem. It is not oriented to any part of the ecosystem, and has a separate conceptual approach to managing the various elements.

Components

A “component” refers to a piece of the Swarm ecosystem. This can be Swarm Storage, the Content Gateway service, PXE booting services, or even the SCS server itself. A component must be registered with the SCS server for it to be brought under the umbrella of SCS management capabilities. Additionally, multiple versions of a component may be registered at the same time, though one may be marked as the currently active version.

Components may additionally provide settings available to be adjusted for a given site. These settings may then be used to populate configuration for running instances of the various components.

 Configuration may reference the settings provided by *any* component registered with SCS, and is not restricted to the settings for a component.

Groups

A “group” refers to a cluster or pool of instances of a given component. The most common example is a Swarm Storage cluster: the cluster is represented as a “group” within SCS. Accordingly, the group’s name is also the name of the cluster. Other installation topologies may involve multiple concurrent pools of Content Gateways:

- A group solely for S3 access
- A group for administrative purposes, outside of the normal data path
- A group providing Content Portal

SCS server also supports the notion of a default group (referred to using the keyword `_default`).

Instances

An “instance” refers to a single running unit of a component. For example, a single Storage node, or a single Content Gateway, or a single SCS server. Each of these represents an instance of the respective component.

Settings

Settings are self-describing values adjusted or modified to suit the needs of a given Swarm site. A setting is associated with the component providing it, and is referenced by name. A setting has a data type, a description, a value, and may have a default value. Additionally, settings may be marked as secure, in which case SCS redacts the value under certain circumstances.

Settings are defined at the component level but may have overrides provided at the group and/or instance level. An instance-level override takes precedence over a group-level override, and if no overrides are given then the base component definition is used.

Templates

Templates and settings are the two sides of the configuration coin. Settings define what values are available for configuration, but templates are what actually pull those values in to a usable form. Each component is unique in the configuration needs: Storage has one configuration file (node.cfg), while Content Gateway has multiple (gateway.cfg, logging.yaml, etc.). Each configuration file may also have a distinct format.

To support the wide variety of configuration needs, components also register templates with SCS server, one for each configuration file an instance may need. These templates refer to settings provided by one or more components, and may also refer to other SCS server information (all known IP addresses of instances of a group within a certain component, for example). The sole formatting "rule" for a template is it must conform to Jinja2 templating rules. Templates are rendered to the usable form when requested by a running instance.

SCS Administration

SCS's CLI (command-line interface) is installed by default on the SCS server and supports common administrative tasks.

- [Getting Help](#)
- [Listing Components](#)
- [Listing Groups](#)
- [Listing Instances](#)
- [Defining the Storage Cluster](#)
 - [Create the Cluster](#)
- [Assigning a Storage Node to a Subcluster](#)
- [Updating a Cluster Setting](#)
- [Updating a Storage Node Setting](#)
- [Resetting a Setting](#)
 - [Instance Level](#)
 - [Group Level](#)
- [Updating Network Settings](#)
 - [DNS Servers](#)
 - [NTP Servers](#)
- [Administrative Credentials](#)
 - [Setting the Administrative User Name](#)
 - [Setting the Administrative Password](#)
 - [Updating CLI Credentials](#)
- [Upgrading Swarm Storage](#)
- [Backing Up SCS](#)
 - [Full Backup](#)
 - [Lightweight Backup](#)

Getting Help

Every command within the CLI offers help. Some examples:

- `scsctl help`
- `scsctl init dhcp help`
- `scsctl repo component add help`

Listing Components

List the components registered with SCS:

```
scsctl repo component list
```

The result list displays active components (including the version that has been marked as active) as well as inactive components (in which no version has been marked as active).

Listing Groups

List the groups for a component:

```
scsctl {component} group list
```

Listing Instances

List the instances within a given group of a component:

```
scsctl {component} instance list --group "{group name}"
```

List the nodes in the Swarm Storage cluster (-d is used to refer to the default group rather than referring to it by name):

```
scsctl storage instance list -d
```

Defining the Storage Cluster

The `storage` component within SCS only allows a single group/cluster to be defined for that site. The name of that cluster is governed by the name assigned to its group within SCS.

Create the Cluster

To create a group for Swarm Storage:

```
scsctl storage group add "{cluster name}"
```

Assigning a Storage Node to a Subcluster

Each node forms a de-facto subcluster if no explicit subcluster assignments are made in Swarm Storage configuration. The Swarm Storage component (`storage`) provides the `node.subcluster` setting as a free-form name that may be assigned to one or more nodes.

The storage process looks at all names assigned to the different nodes and forms them into groups, which can then be used to determine how object replica distribution and protection are handled. The nodes may be grouped using subclusters in any way needed to achieve the desired replica/fail-over paradigm.

Update the subcluster for a storage node:

```
scsctl storage config set -d --instance "{instance name/ID}" "node.subcluster={subcluster name}"
```

Updating a Cluster Setting

Update a cluster setting for Swarm Storage:

```
scsctl storage config set -d "{setting name}={setting value}"
```

Some specific examples:

```
scsctl storage config set -d "policy.versioning=allowed"  
scsctl storage config set -d "policy.eCEncoding=4:2"
```

Updating a Storage Node Setting

Update a cluster setting for Swarm Storage:

```
scsctl storage config set -d --instance "{instance name/ID}" "{setting name}={setting value}"
```

Some specific examples:

```
scsctl storage config set -d --instance "{instance name/ID}" "ec.protectionLevel=node"  
scsctl storage config set -d --instance "{instance name/ID}" "feeds.maxMem=500000"
```

Resetting a Setting

Removing a setting override means that the value for the setting is inherited from a higher scope. Removing an instance-level override means that the value for the setting is obtained from either the group (if a group-level override has been set) or component level. Removing a group-level override has no influence on any existing instance-level overrides that may exist within that group.

Instance Level

Reset an instance-level override:

```
scsctl {component} config unset --group "{group name}" --instance "{instance name/ID}" "{setting
```

Group Level

Reset a group-level override:

```
scsctl {component} config unset --group "{group name}" "{setting name}"
```

Updating Network Settings

Shared network settings, such as DNS information and NTP time sources, may be updated as the need arises.

DNS Servers

Update the list of DNS servers (specified as comma- or space-delimited list):

```
scsctl network_boot config set -d "network.dnsServers={new DNS servers}"
```

This also requires that the DHCP server be updated so the setting can be made available to booting Storage nodes.

```
scsctl init dhcp {reserved ranges}
```

NTP Servers

Update the list of NTP servers (specified as comma- or space-delimited list):

```
scsctl platform config set -d "network.ntpServers={new NTP servers}"
```

This also requires that the DHCP server be updated so the setting can be made available to booting Storage nodes.

```
scsctl init dhcp {reserved ranges}
```

Administrative Credentials

The SCS server maintains an administrator user that has full rights within the Swarm site. This user also serves as the administrative user within the Swarm Storage management API. Credentials may be updated at any time, and updates are pushed to the Storage cluster to guarantee the two use the same credentials.

i Logging in to the CLI is required to perform these operations if administrative credentials have already been set within SCS. The CLI credentials need to be updated once either the user name or password has changed.

Setting the Administrative User Name

Update the administrative user name:

```
scsctl platform config set -d "admin.userName={new user name}"
```

Setting the Administrative Password

Update the administrative password:

```
scsctl platform config set -d "admin.password={new password}"
```

Updating CLI Credentials

The CLI requires knowing the administrative credentials to perform operations against the SCS server. To set these credentials:

```
scsctl auth login --user "{administrative user name}"
```

The CLI then securely prompts for the administrative password and proceed with authentication.

Upgrading Swarm Storage

Obtain a new component bundle for the desired version from DataCore to upgrade the Swarm Storage software of a running cluster. Copy it to the SCS server and run the following command to register it with SCS:

```
scsctl repo component add -f "{path to component bundle}"
```

Verify the presence in the list of available versions once the new version has been added:

```
scsctl storage software list
```

 It should be automatically marked as active if this is the first time Swarm Storage software has been registered with SCS. The following step may be skipped otherwise proceed with activation if so.

It has been successfully registered if the new version is in the list. It is not used for booting nodes and the current active version is used. Mark it as active to complete the upgrade:

```
scsctl storage software activate "{new version}"
```

 Activating a version means that any nodes that reboot use the binaries for the new version. Do not complete this step until ready to proceed with the upgrade.

Backing Up SCS

SCS allows a full backup of all components, configurations, settings overrides, and binaries for support and maintenance purposes. The CLI must be logged in since this backup includes values for settings marked as “secure”.

Full Backup

Obtain a full backup of all data:

```
scsctl backup create --output "{path to output backup file}"
```

Lightweight Backup

Obtain a “lightweight” backup that excludes repo data (binaries, etc.):

```
scsctl backup create --no-repo --output "{path to output backup file}"
```

Swarm Content Gateway

- [Managing Dynamic Features](#)
- [Content Metering](#)
- [Gateway Operations](#)
- [Replicating Domains to Other Clusters](#)
- [Content Gateway Concepts](#)
- [Upgrading Gateway](#)
- [Content Gateway Authentication](#)
- [Gateway Troubleshooting](#)
- [Object Locking](#)

Managing Dynamic Features

Gateway has infrastructural support for optional features and extensions (such as [Video Clipping for Partial File Restore](#)) that can be drop into a Swarm implementation as desired. (v6.1)

- [Installing Dynamic Features](#)
- [Metrics for Feature Usage](#)
- [Audit Logging for Features](#)

Installing Dynamic Features

After obtaining the optional package for a feature, install it on the Gateways, along with any supporting frameworks needed, such as FFmpeg, which DataCore provides preconfigured within a Docker container.

Best practice

Use Docker and install DataCore's provided container; this avoids dealing with third-party repository choices, dependencies, and resource reconfiguration required to protect Gateway's performance. Installing without the container requires assistance from DataCore Support.

1. Copy the package for the feature to a server running Content Gateway:
`caringo-FEATURE-VERSION.noarch.rpm`
2. Upgrade to RHEL/CentOS 7 (required by Docker) if the server is running RHEL/CentOS 6.x.
3. Upgrade Gateway (versions 6.1 and higher support drop-in features such as video clipping) if running Content Gateway 6.0 or earlier. See [Gateway Installation](#).
4. Upgrade Content UI to support this feature if running version 6.1 or earlier. See [Content UI Installation](#).
5. Install the package.

```
yum install caringo-FEATURE-VERSION.noarch.rpm
```

The installation creates a `features.d` directory under `/etc/caringo/cloudgateway/`, which is where Gateway detects optional, dynamic features.

6. Install any additional frameworks, such as multimedia support via FFmpeg, which DataCore provides preconfigured in a Docker container. From an Internet-connected machine:
 - a. Check whether Docker is installed: `docker info`
Else, install the Docker package, start the daemon, check its status, and enable it system-wide. See docs.docker.com/install/linux/docker-ce/centos/
 - b. Verify Docker by running a container test:

```
docker run hello-world
```
 - c. Load the provided container:

```
docker load < caringo-gateway-VERSION.feature.FEATURE.via.docker.VERSION.x86_64
```
7. Gateway creates the following directory for spooling content: `/var/spool/caringo/cloudgateway/features`
Gateway refuses to start if it cannot create this directory.
8. In setting `iptables` rules, Docker closes external TCP access to port 80 (although ping and ssh work); explicitly open port 80 again so clients can access Gateway.

```

firewall-cmd --add-port=80/tcp --permanent
firewall-cmd --reload
systemctl restart docker
    
```

9. Repeat the above process on any remaining Content Gateway servers.
10. Restart the Gateway(s).
 On reboot, Content Gateway detects the feature; on a page refresh, Content UI displays the additional functionality, such as the clipping control for video content.

Metrics for Feature Usage

Dynamic features include a set of metrics providing visibility in to how each feature is being used. (v6.2)

These are the [Prometheus Node Exporter and Grafana](#) metrics displaying dynamic features:

```
curl http://GATEWAY:9100/metrics -s | grep -i feature
```

caringo_gateway_feature_install_count	How many dynamic features are installed currently.
caringo_gateway_feature_invocation_count	How many times was the feature called, since the last Gateway restart.
caringo_gateway_feature_invocation_latency	How much time, on average, did successful calls for that feature take, since the last Gateway restart. For video clipping, this varies relative to the size of the clips being created.
caringo_gateway_feature_errors_count	How many times did the feature call fail, since the last Gateway restart.

See [Prometheus Node Exporter and Grafana](#).

Audit Logging for Features

Each dynamic feature logs operations to provide auditing. When creating a video clip, for example, Gateway handles it asynchronously and acknowledges the request with an INVOKE message, which appears *first* in the audit log. That acknowledgement references the future JSON result object. When that JSON result later posts, it reports the outcome of the clipping request, such as the FFmpeg exit code and the duration, capturing the same information received on the response if it is synchronous.

```

2019-08-22 14:32:04,991 INFO [F38143E84D3EC62E] 2 192.168.1.154 192.168.1.154 Feature:videoclippi
2019-08-22 14:32:15,022 INFO [F38143E84D3EC62E] 2 192.168.1.154 192.168.1.154 Feature:videoclippi
2019-08-22 14:32:15,061 INFO [F38143E84D3EC62E] 2 192.168.1.154 192.168.1.154 Feature:videoclippi
    
```

All JSON result objects are temporary, by default: they are created with a lifepoint that triggers deletion after 5 days. Change the default in the [Gateway Configuration](#), `gateway.cfg ([dynamic_features] responseObjectLifetime=5)`.

See [Gateway Audit Logging](#).

Content Metering

- [Configuring Metering](#)
- [Metering Statistics](#)
- [Metering API](#)
- [Example Metering Requests](#)
- [Index Generation for Metering](#)
- [Retaining Data for De-provisioned Resources](#)

Gateway's metering is a scalable, flexible, integrated usage metering solution that makes use of Elasticsearch for data storage, management, and analysis. Configure usage metering to send batched storage and network statistics to the Elasticsearch server at whatever interval needed. Access the metering data by querying Elasticsearch directly or through the [Content Management API](#). Metering gathers the data needed to manage the business of the organization:

- Current usage numbers allow evaluating usage quotas.
- Usage over time allows billing generation.
- Historical usage queries populate graphs for easy monitoring from the dashboard.



Note

Metering replaces the legacy CSMeter package (`csmeter` and `cshistory` utilities), which is deprecated and no longer included with the Gateway.

Configuring Metering

Metering requires minimal configuration to implement:

- **metering.enabled** (disabled by default)
- **storage_cluster.indexerHosts** (must be defined for metering)

Gateway includes other configuration parameters that are specific to controlling metering for special cases.

See [Gateway Configuration](#).

Metering Statistics

Gateway emits two types of usage statistics: *storage* and *network*.

Type	Statistic	Description	Notes
Storage	bytesSize	Sums the bytes of <i>content</i> (logical objects, including versions) uploaded for storage in the context. Summing the individual <code>Content-Length</code> headers of the objects provides this value.	Swarm storage usage statistics are reported by bucket, domain, and tenant. Untenanted domains are grouped into a synthesized "_system" tenant. The <i>context</i> is a domain or a domain and bucket.

	<p>bytesStored</p> <p>Sums the bytes of <i>space used on disk</i> by all Swarm objects in the context, including all replicas and erasure-coded segments. This is also commonly called raw storage.</p> <p>This statistic yields a value that is the expected number of replicas in the cluster and does not account for temporary under- or over-replication that may exist in the cluster.</p>	The absence of a bucket returns the storage used by unnamed objects.
	<p>objectsStored</p> <p>Counts the number of unique objects being stored in the context.</p>	
Network	<p>bytesIn</p> <p>Sums the bandwidth usages from clients to the Gateway.</p>	Network usage is reported at the context to which the requests are made; the particular bucket or domain .
	<p>bytesOut</p> <p>Sums the bandwidth usages from the Gateway to clients.</p>	
	<p>opCount</p> <p>Counts the number of client operations.</p>	Network usage only includes storage operations and excludes Management API requests.

Metering API

The API for metering is part of the Gateway's [Content Management API](#).

Request

Each tenant, domain, and bucket has a subresource prefix, **meter**:

- Tenant **t1**: `/_admin/manage/tenants/t1/meter/`
- Domain **d1**: `/_admin/manage/tenants/t1/domains/d1/meter/`
- Bucket **b1**: `/_admin/manage/tenants/t1/domains/d1/buckets/b1/meter/`

Under **meter**, this is the endpoint for the specific context (tenant, domain, bucket):

Metering endpoint for context

```
meter/usage/{metric}
?from={startDate}
&to={endDate}
&groupBy={groupBy}
```

Value	Required	Case	
-------	----------	------	--

{metric}	Yes	Case-sensitive	<p>Specifies which metric to analyze:</p> <ul style="list-style-type: none"> • bytesIn (from client to Swarm) • bytesOut (from Swarm to client) • bytesSize (sum of logical objects' Content-Length values) • bytesStored (sum of physical disk storage consumed) • objectsStored (number of logical objects) • opCount (operation count, minus Management API requests) <p>These requests are supported for point-in-time (current) queries for untenanted objects: (v6.2)</p> <ul style="list-style-type: none"> • bytesSize/untenanted (sum of logical objects' Content-Length values) • bytesStored/untenanted (sum of physical disk storage consumed) • objectsStored/untenanted (number of logical objects)
{startDate}	Yes		<p>YYYY-MM-DDT00:00Z YYYY-MM-DDThh:mmZ YYYY-MM-DDThh:mm:ssZ</p>
{endDate}	Yes		<p>YYYY-MM-DDT00:00Z YYYY-MM-DDThh:mmZ YYYY-MM-DDThh:mm:ssZ</p> <p>Must be later than {startDate}.</p>
{groupBy}	No	Case-sensitive	<p>Specifies which time increment to group by:</p> <ul style="list-style-type: none"> • hour • day • <i>unspecified</i> - no grouping: date range is collapsed to a single value <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>Group by aggregates metrics using <i>average</i> for storage metrics and <i>sum</i> for network metrics.</p> </div>

Add `/children` to fetch a metric for the children of a context (either the tenant's domains or the domain's buckets):

Metering endpoint for child of context

```
meter/usage/{metric}/children
?from={startDate}
&to={endDate}
&groupBy={groupBy}
```

Run a point-in-time query specific to storage metrics:

Point-in-time storage metrics

```
meter/usage/bytesSize/current
meter/usage/bytesStored/current
meter/usage/objectsStored/current
```

```
meter/usage/bytesSize/current/children
meter/usage/bytesStored/current/children
meter/usage/objectsStored/current/children
```

```
meter/usage/bytesSize/untenanted/current
meter/usage/bytesStored/untenanted/current
meter/usage/objectsStored/untenanted/current
```

Note

For all `current` metrics, no date range is required and grouping is not applicable.

Response

The response to a query is an array of objects ("rows"), with fields that correspond to the data for each entry. These are the possible fields:

tenant domain bucket	The name of the applicable tenant, domain, or bucket for the object. Untenanted domains are grouped within the "_system" tenant name. Unnamed objects in a domain are grouped within an empty string ("") bucket name. The domain can also be an empty string, recording requests at the tenant level outside of any domain. If the domain or bucket had activity during the requested timeframe, but the name is not available because it has been deleted, the UUID is returned instead. The UUID corresponds to the former domain or bucket's <code>Castor-System-Alias</code> value.
bytesIn bytesOut bytesSize bytesStored objectsStored opCount	The value for the metric requested, which corresponds to the {metric} from the request.
timestamp	For queries grouped by time, the timestamp for a given time grouping. the timestamp identifies which of those 7 days relates to each result if grouping by day across a week of time.

An empty list ("[]") is returned if no records exist for the query range. When fetching the children of a context, if a child has no data for the query range, the record is excluded from the response.

Note

Using `/children` and a time grouping together may result in additional rows to express each time/child combination, as RDBMS queries with multiple GROUP BY arguments return separate rows per every combination.

Example Metering Requests

Example request/response

Following is a result from a query for `/children` that uses a `day` grouping. The target of this query is the domain `domain1`, which belongs to tenant `tenant1`.

Note: the results are for the *children* of the domain, which are the buckets (as opposed to the children of a tenant, which are the domains):

Example metering request/response

```
GET /_admin/manage/tenants/tenant1/domains/domain1/meter/usage/bytesIn/children
?from=2015-07-01T00:00Z&to=2015-07-03T00:00Z&groupBy=day
```

```
[
  {
    tenant: "tenant1",
    domain: "domain1",
    timestamp: "2015-07-01T00:00:00.000Z",
    bucket: "research",
    bytesIn: 27277
  }, {
    tenant: "tenant1",
    domain: "domain1",
    timestamp: "2015-07-01T00:00:00.000Z",
    bucket: "archive",
    bytesIn: 18771
  }, {
    tenant: "tenant1",
    domain: "domain1",
    timestamp: "2015-07-02T00:00:00.000Z",
    bucket: "research",
    bytesIn: 27855
  }, {
    tenant: "tenant1",
    domain: "domain1",
    timestamp: "2015-07-02T00:00:00.000Z",
    bucket: "archive",
    bytesIn: 19645
  }
]
```

Common billing queries

These are queries that are common when integrating with billing systems where charges for bandwidth in/out and storage are calculated at the end of a calendar month. In these examples, the period being queried is Midnight 2016-06-01 UTC through Midnight 2016-07-01 UTC. Note: the storage numbers are the average storage over the month while the bandwidth is the total at the end of the month.

System tenant raw storage by domain

```
GET /_admin/manage/tenants/_system/meter/usage/bytesStored/children
?from=2016-06-01T00:00:00Z&to=2016-07-01T00:00:00Z
```

Tenant 'bravo' raw storage

```
GET /_admin/manage/tenants/bravo/meter/usage/bytesStored
?from=2016-06-01T00:00:00Z&to=2016-07-01T00:00:00Z
```

Domain 'xray.example.com' raw storage

```
GET /_admin/manage/tenants/delta/domains/xray.example.com/meter/usage/bytesStored
?from=2016-06-01T00:00:00Z&to=2016-07-01T00:00:00Z
```

Bandwidth IN for domains of tenant 'tango'

```
GET /_admin/manage/tenants/tango/meter/usage/bytesIn/children
?from=2016-06-01T00:00:00Z&to=2016-07-01T00:00:00Z
```

Bandwidth OUT for domains of tenant 'tango'

```
GET /_admin/manage/tenants/tango/meter/usage/bytesOut/children
?from=2016-06-01T00:00:00Z&to=2016-07-01T00:00:00Z
```

Raw storage for domain 'uniform.example.com' by bucket

```
GET /_admin/manage/tenants/tango/domains/uniform.example.com/meter/usage/bytesStored/children
?from=2016-06-01T00:00:00Z&to=2016-07-01T00:00:00Z
```

Logical storage for domain 'uniform.example.com' by bucket

```
GET /_admin/manage/tenants/tango/domains/uniform.example.com/meter/usage/bytesSize/children
?from=2016-06-01T00:00:00Z&to=2016-07-01T00:00:00Z
```

Index Generation for Metering

Metering uses a different index for each day, making it efficient to expire old data. The daily index is created by utilizing Elasticsearch's index alias to combine the records into a queryable whole. To support this, each gateway runs a daily maintenance task that deletes any indices older than the retention period and adds the new daily index to the alias. This maintenance is scheduled with offsets to avoid having multiple gateways performing the task simultaneously.



Note

Statistics are batched for a period of time. The date samples are assigned is the date when they are *written*, not when they are *collected*. A new index for the new day is created automatically.

Retaining Data for De-provisioned Resources

Do not inadvertently lose access to metering data when de-provisioning a tenant or a domain. There is no longer a Content Management API path to which to refer for metering queries when a tenant or domain is removed.

Best practice - Retain the historical metering data for each decommissioned entity:

1. Change ownership of the tenant or domain to be an admin user.
2. Update the Policy to remove non-admin access.
3. Purge all objects being stored in the decommissioned domain (or all domains for a decommissioned tenant).
4. **Important:** Do not delete the empty tenant or domain; retain it as is.

Delete the decommissioned tenant or domain once no longer needed to retain the historical usage data. The domain or bucket name is not available but it may still be returned in metering queries covering an earlier time once deleted. The UUID is returned instead of the name.

Gateway Operations

- [Service Control and Status](#)
 - [RHEL/CentOS 7](#)
- [Java Runtime Parameters](#)
 - [Default Java Process Settings](#)
- [Server Firewall](#)
 - [RHEL/CentOS 7](#)
- [Create Domains and Buckets](#)

Service Control and Status

Use the following commands to manually control the start-up and shutdown of the Gateway service and to query the running status:

RHEL/CentOS 7

```
systemctl start cloudgateway
systemctl stop cloudgateway
systemctl status cloudgateway
```

Java Runtime Parameters

The `/etc/sysconfig/cloudgateway` file contains the Java process memory settings and the maximum number of open file descriptors for Gateway.

Default Java Process Settings

```
HEAP_MIN=1024m
HEAP_MAX=1024m
MAX_OPEN_FDS=25000
```

Memory: While the default JVM heap memory settings work well for a small deployment, increase them for larger deployments handling a large transaction load for multiple tenants. The JVM heap memory is utilized for caching frequently used objects, authentication results, and other operational data.

Maximum file descriptors: While the default maximum number of open file descriptors works well for a small deployment, increase the limit for larger deployments handling a large transaction load for multiple tenants. Network socket connections for the upstream clients plus the back-end connection pool comprises the majority of the open file descriptors during Gateway operations.

Server Firewall

The firewall rules from the default RHEL/CentOS installation need to be changed to allow inbound client access to Gateway. Adjust the IPTABLES rules to allow inbound access for each front-end protocol or disable IPTABLES entirely. Execute the following commands to disable the operating system's firewall:

RHEL/CentOS 7

```
systemctl disable firewalld
systemctl stop firewalld
```

While it is valid to use IPTABLES in conjunction with Gateway, the service startup script issues a notice if IPTABLES are enabled as a reminder since using them can be a source of confusion if inbound traffic to Gateway is blocked. Ignore this startup notice if the inbound rules to allow access are customized.

Create Domains and Buckets

See [Configuring Domains](#) and [Configuring Buckets](#) to create and manage domains and buckets from the Content UI.

See [Manually Creating and Renaming Domains](#) if manual creation is needed (from the command line).

Replicating Domains to Other Clusters

Replication of domains between Swarm clusters provides for disaster recovery and locality of access to content. Many replication strategies are supported by Swarm including single direction roll-up, multi-master, and cascading topographies. This section focuses on the content replicated to host storage domains in multiple clusters.

Domain creation

Regardless of the replication strategy selected, it is crucial a domain, whether an administrative domain or a storage domain, is only created once.

A different domain sharing the same name is created if a domain with the same name is created using an SCSP operation or with the `initgateway` command in multiple Swarm clusters. This leads to incorrect results if the different domains are ever replicated into the same cluster due to the name collision. Create a domain name one time and use a Replication Feed to copy it to other clusters.

Gateway adminDomain

Do not create the same domain in two clusters: always create it in the source cluster and then *replicate* it to the target. A Gateway must use an independent adminDomain, at least temporarily, if the Gateway is in front of the target cluster. (CLOUD-2785)

Verify the administrative domain is available in the other cluster if intending to use a storage domain within a cluster other than the one in which it was initially created. To use a storage domain means client requests (read, write, delete) are performed in the cluster. Replication of the administrative domain is unnecessary if the storage domain is replicated to another cluster purely for Disaster Recovery (DR) and client requests are ever sent to it. the `initgateway` command must be used in one cluster and then use Replication Feeds to duplicate the administrative domain into all other clusters.

Required permission

Swarm needs to make a "GET /" request during replication to check the Swarm cluster name and version when replicating through Gateway.

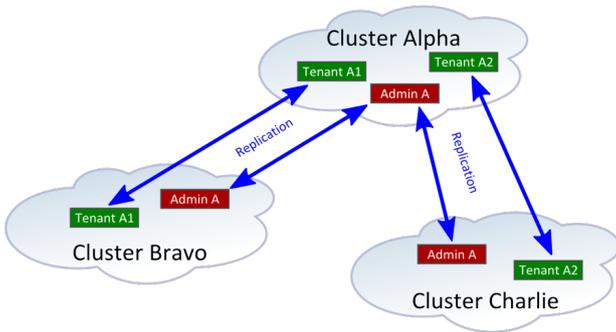
Add a **policy.json** rule providing "anonymous" permission to "GET /" (GetObject) to enable Swarm to perform this check:

```
{
  "Effect": "Allow",
  "Sid": "Swarm Node Status",
  "Principal": {
    "anonymous": [
      "*"
    ]
  },
  "Action": [
    "GetObject"
  ],
  "Resource": "/"
},
```

See [Policy Document](#).

Example replication

The following diagram shows three Swarm storage clusters, storage domains A1 and A2, and an administrative domain A. Domains A, A1, and A2 are all initially created in Cluster Alpha. Remote replication was then configured to mirror domains A and A1 to Cluster Bravo. Additionally, domains A and A2 are configured to mirror to Cluster Charlie.

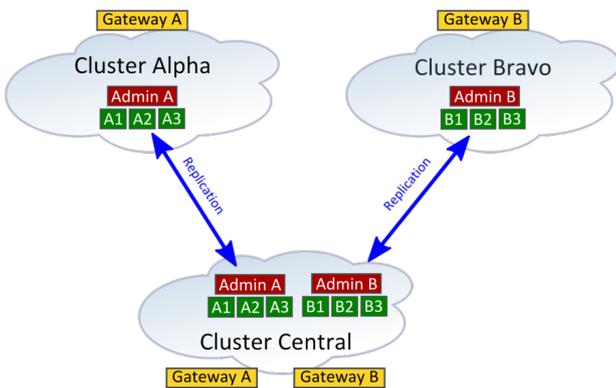


Each cluster can support a Content Gateway configured to use domain A as the administrative domain by configuring the domains to mirror, bi-directional replication. With Gateways deployed in all three clusters and all using administrative domain A, clients may access storage domain A1 from Cluster Alpha and Cluster Bravo. Clients may access storage domain A2 from Cluster Alpha and Cluster Charlie.

Content is available for reading from all clusters to which it is mirrored and content changes (create, update, or delete) are propagated to the other clusters by mirroring the storage domains. Notice the administrative domain for a storage domain must be mirrored to any cluster where the clients access the

storage domain. This is required so the IDSYS, Policy, XFORM, authentication tokens, and other tracking information used to manage the storage domain are available to the Gateways running in the other clusters. Although replication of the administrative domain is not required if clients do not access a storage domain from another cluster, it is recommended to simplify DR if it becomes necessary to restore.

It is also possible to replicate storage domains with different administrative domains into the same Swarm storage cluster while providing client access to all storage domains.



Client access is enabled by deploying a Content Gateway server for each set of storage domains when a Swarm storage cluster hosts multiple sets of storage domains and the corresponding administrative domains. Although a Gateway uses one administrative domain, it can access any storage domain associated with the administrative domain as long as the storage domain exists in the local Swarm storage cluster. The previous diagram shows storage domains A1, A2, and A3 accessed through Gateway A in both Cluster Alpha and Cluster Central. The deployment of Gateway A in both clusters makes use of the mirrored administrative domain A to manage the mirrored storage domains.

The client use cases and application architecture must account for replication latency when remote replication is being used as described. Following the creation, update, or deletion of content in one domain, there is a time delay

before the change is observed within another cluster. This latency depends on the inter-cluster bandwidth and the total replication workload between the Swarm clusters.

Content Gateway Concepts

Content Gateway is the key to implementing a cloud storage service that is both successful and secure. Combined with Swarm, Content Gateway provides everything needed to deliver private cloud storage services secure within your data center or a public cloud storage service that can compete with Amazon S3. These are essential features of Content Gateway:

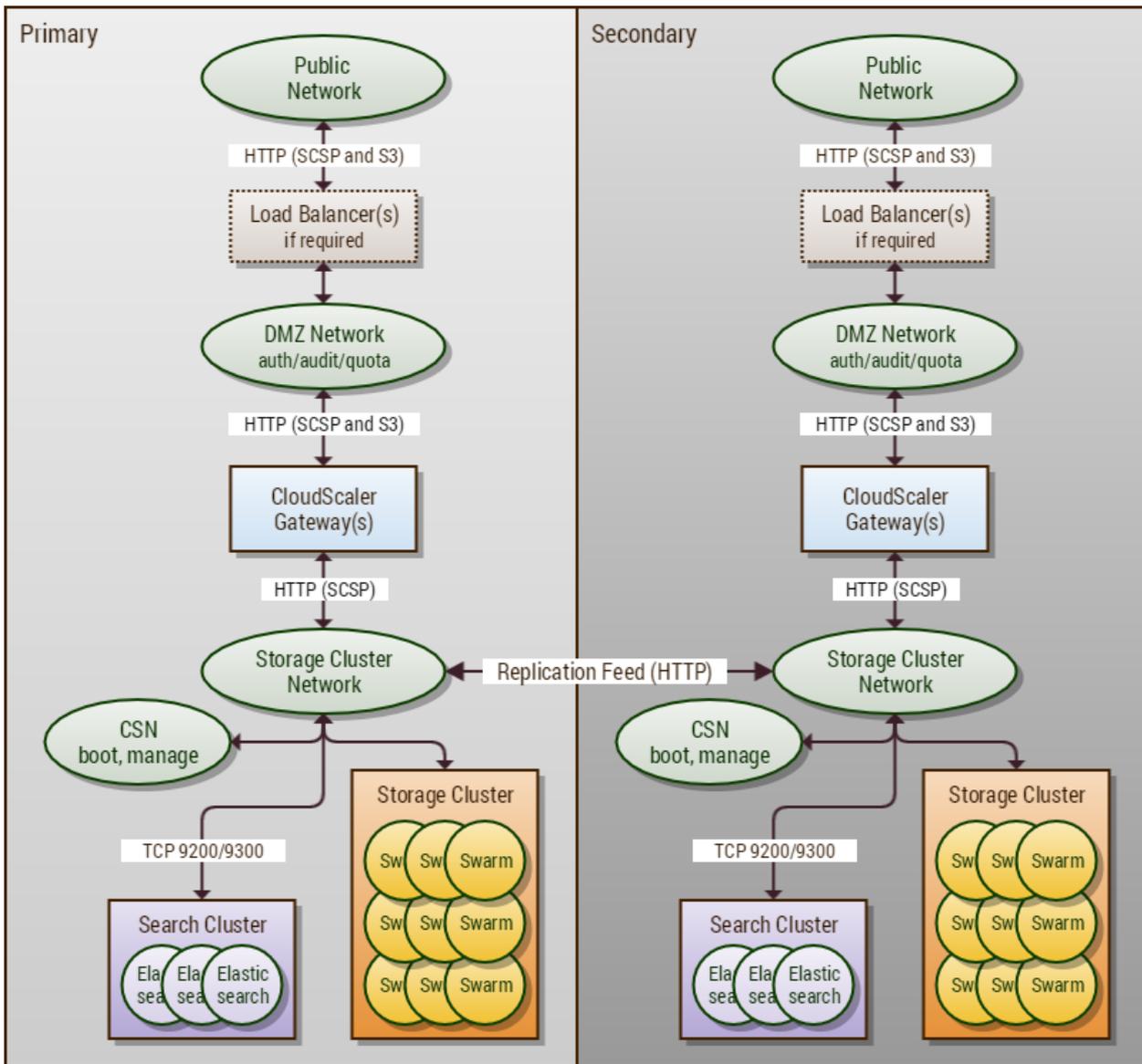
- **Works with existing authentication systems.** Use LDAP, Active Directory, and Linux PAM authentication for integration into existing corporate identity management systems. Additionally, the system supports token-based authentication for pre-validated access with an optional automatic expiration.
- **Granular control of content access.** A robust access control mechanism allows for coarse to fine-grained control over access to content within the system.
- **Robust administration.** Administration is managed through the web-based Content Portal or programmatically through an open and documented management API.
- **Metering of usage.** Provides historical records of storage and bandwidth usage within the system. The metering data is presented to admins and end-users through the graphical charting in the Content Portal and made available through the management API for integration with external systems and applications.
- **Audit logging.** Administrators can access a data feed that includes details about the all requests to the system that includes the success /rejection response, the user making the request, and the duration of the request. The data feed is designed for machine parsing so it can be used for access audits, API request analysis, and SLA reporting.
- **Flexible naming schemes with no bucket limits.** Easy storage management and flexible bucket naming is enabled by lightweight tenant and domain creation and allocation.
- **Uses your domain.** Access and deliver content using your corporate domain name. Virtual hosting of buckets is also supported through the S3 API.
- **Amazon S3 API support.** Provides instant compatibility with a broad range of applications and libraries that use Amazon's S3 service.
- [Gateway Architecture](#)
- [Role of the Gateway](#)
- [Content Gateway Components](#)
- [Application Concepts](#)
- [Service Proxy](#)

Gateway Architecture

The Content Gateway involves the following components:

- Client applications that access the object storage cloud via RESTful HTTP calls
- Optional front-end load balancing and/or firewall appliances
- Content Gateway server(s)
- Elasticsearch server(s)
- Swarm Storage cluster

This is an example deployment architecture for the Content Gateway components:



Role of the Gateway

The Content Gateway is a lightweight, web-scale software application used by organizations who need to deploy massively scalable, secure, multi-tenant object storage clouds.

The Gateway works as a reverse-proxy in front of the Swarm storage cluster and extends the RESTful object storage API with the following capabilities:

- S3 object storage protocol
- User and group authentication with external identity management systems
- Access control policies
- Automatic metadata transformation
- Usage metering
- Audit logging

Content Gateway also serves enterprises who need to provide a private storage cloud for business units and Managed Service Providers and Cloud Service Providers that want to offer public cloud storage as a service.

Content Gateway Components

The Content Gateway platform architecture is comprised of hardware and software components.

- [Client applications and users](#)
- [Load balancer](#)
- [Protocol personalities](#)
- [Gateway](#)
- [Metadata search servers](#)
- [Swarm storage cluster](#)
- [Remote replication](#)

Client applications and users

Client applications use the Gateway over a network and communicate via the SCSP storage protocol or another protocol such as S3 translated by a protocol personality. Client applications include common web browsers such as Firefox, Chrome, or Safari or they can be software from third-party ISVs or custom in-house software developed by clients. Users are people utilizing client application software communicating with Gateway. The implication is users are making use of one or more client applications to interact with Gateway.

Load balancer

Load balancer appliances are a common method for automatically distributing client requests across all Gateways and for excluding Gateways offline due to failure or maintenance when deploying more than one Gateway. Load balancers may optionally implement upstream features such as SSL/TLS end-point termination, protocol firewall rules, quality of service, and geographic traffic management.

The Gateway appears as a normal web proxy to the upstream load balancers and, since the Gateways are stateless, it is not necessary to implement session affinity on the load balancers. Load balancing schemes such as weighted round-robin that prefer to dispatch to the most responsive Gateways are a good choice.

The load-balancing layer is optional for Gateway, and, while hardware appliance load balancers can be well-suited for sites with heavy traffic and sophisticated operational requirements, it is also possible to implement this layer using virtual machines or modest hardware running open-source software. For example, the *Pound* reverse proxy running on Linux provides transport encryption and load balancing with Layer 7 inspection capabilities.

Protocol personalities

Protocol personalities are optional protocol translators allowing client applications to communicate using a different storage protocol than SCSP. By translating communications in this manner, all client applications, regardless of the protocol they use, share the same content in the back-end cluster and they utilize a common authentication/ACL scheme. An analogy for this universal storage protocol access is ODBC for database communications.

The SCSP and S3 protocol handling is implemented natively within the Gateway. Third-party or user-developed personalities may also be added to, or in front of, the Gateway server.

Gateway

The Gateway is a value-added front-end for the Swarm storage cluster. At the core it is a stateless reverse proxy deployed in an $n+1$ configuration for horizontal scaling and high-availability. The value-added features provided by the Gateway enhance the Swarm SCSP client protocol and provide storage management and protection for the back-end storage cluster.

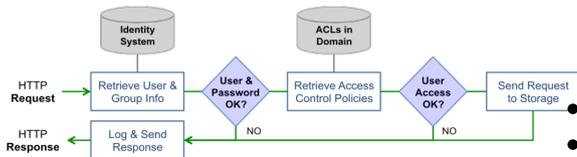
These value-added features include:

- Authentication for users and access control for content
- Usage metering of storage and bandwidth
- Audit logging for client operations
- Automatic object metadata transformation rules
- Cluster node pool management for load balancing and handling offline nodes
- Reverse proxy to handle SCSP redirects locally to optimize and simplify client communications
- Token-based authentication
- Multi-part MIME uploads

The Gateway is a Java software component running within a Jetty servlet container and provides front-end HTTP web services to client applications. The Gateway servers are typically deployed with dual-homed network interfaces to provide proxy services between a front-end client network and a private, back-end storage network.

The Gateway provides protocol isolation and performs SCSP protocol inspection for the incoming client storage requests before passing them along to the storage cluster. This allows for, among other things, the implementation of business rules for content metadata, access control and administrative override for tenant content, and audit/billing event logging.

The Gateway implements the following authentication and request authorization logic:



The following Swarm features cannot be used when communicating *through* the Gateway when using the Gateway front-end for Swarm:

- Integrity Seal hash-type upgrades
- Trailing Content-MD5 headers
- [deprecated] Swarm legacy auth/auth mechanism
- Swarm legal hold mechanism

Metadata search servers

Elasticsearch servers provide a NoSQL data query engine enabling metadata searching with the Swarm storage cluster. The query engine software allows for $n+1$ deployments providing horizontal scalability for load sharing and high availability.

See [Elasticsearch for Swarm](#).

Swarm storage cluster

The Swarm storage cluster provides scalable, object storage engine for the Gateway platform. The storage engine consists of standard x86 hardware that manage and protect storage for multiple tenants.

See [Storage Implementation](#) and [Swarm Storage Cluster](#).

Remote replication

The Swarm replication feeds between clusters can be used with Gateway in the following deployment scenarios:

- Clusters communicate directly with each without passing through Gateway
- Gateway acts as a front-end reverse proxy for a cluster

Direct communication between the clusters happens through internal routing rules between the storage networks or over a VPN connection between the storage networks. The key aspect of this communication is no inter-cluster traffic passes through the Gateway.

The `allowSwarmAdminIP` setting must be configured in the `[scsp]` section of the Gateway's configuration file if the Gateway is to act as a front-end reverse proxy for a storage cluster that is the **target** of a Swarm replication feed from another cluster. The value is the IP address list or prefix of every replication source contacting this Gateway.

Application Concepts

The Gateway offers developers an integration platform that adds many valuable features to the native Swarm storage API. These features include:

- A multi-tenant framework that provides several levels of control and delegation
- Choice of two object storage APIs: SCSP, S3
- A service provisioning and management API

Object storage APIs

Gateway provides developers the freedom to choose between the **Swarm Storage SCSP** object storage protocol and the **Amazon S3** object storage protocol. Gateway allows both of these protocols to share the back-end Swarm cluster and even the same content. Additionally, Gateway provides enhancements to both object storage protocols that allow for geographically distributed, multi-tenant storage clouds.

S3 and SCSP are the Storage APIs offered through the Gateway. Developers accustomed with Amazon S3 development can continue to use tools, libraries, and experience and immediately begin using Swarm in the existing environment. The Swarm SCSP object storage protocol offers advantages over Amazon S3 in the area of content protection controls, time-based content policies, metadata searching capabilities, and an additional object type: unnamed objects.

- The SCSP object storage protocol is documented in [Storage SCSP Development](#).
- The S3 object storage protocol implemented by Gateway is documented primarily by Amazon AWS and specific integration topics are covered in the [S3 Protocol Interface](#).

Multi-tenant framework

While Swarm already provides multi-tenant separation of content, Gateway builds upon that foundation by formally defining scopes within the storage system to provide developers with a proven framework for organizing and managing a cloud storage system. These are the scopes defined by Gateway:

Note: while Swarm defines the role of owner, role based access control (RBAC) definitions can be created with varying to sophistication as required for the organization using Gateway's access control policies. The Cluster, Tenant, Domain and Bucket "admins" shown in the diagram above are common roles that can be used but are not required or hard-coded into the system. For the purpose of explaining the scopes and purposes, these roles are assumed to be used in the system.

- **Root Scope.** The root scope exists on the Gateway servers' file systems as the configuration information necessary to bootstrap the cloud storage system. It contains the top-level definition of the identity management system and the overall access control policy for the entire cluster. The Gateway system administrations manage the resources at this level through standard Linux administration tools.
- **Cluster Scope.** The cluster scope is the top-level control point within the object storage system. The cluster administrations operating in this scope are the super users within the cloud storage system and have the ability to create and access all content within the system. Through the Content Portal or using management API calls, they create lower-level scopes, such as tenants and storage domains, and they can delegate management duties to those lower-level scopes to less privileged users.
- **Tenant Scope.** The tenant scope is a formalized concept that exists within Gateway and not within Swarm. A tenant is a hierarchy that owns one or more storage domains. Each tenant scope can define a separate identity management system so users and groups within them are separated from those in other tenants. The tenant administrators have the ability to create and access storage domains on behalf of the tenant and they can delegate management duties for the storage domains they create. The tenant scope does not store end-user data; it is a meta store for information about the tenant, users, and storage domains.

- **Domain Scope.** The domain scope is directly tied to a Swarm storage domain and is where end-user data is kept. The SCSP and S3 storage protocols create and use data within the domain scope. While the domain scope can inherit user and group identity information from its tenant, it also has the ability to define its own identity management system. The domain administrators can create and access all content within the storage domain. They can optionally delegate control of storage buckets to individual users or groups.
- **Bucket Scope.** The bucket scope is directly tied to a bucket that exists within the Swarm storage domain. While access control policies can be defined for every bucket, there is no option for an identity management system definition at the bucket scope. All buckets with a domain share the domain's identity management system definition.
- **IDM.** Identity management system connection information is stored within IDSYS objects and they are the source of user and group information and the authentication system.
- **Access Policy.** The Policy objects contain the rules for access control to content within the system. This includes control of all operations through the Storage API and the Management API. Policies are associated with every scope within the storage system.

Management API

Separate from the Storage API for end-user content, the Gateway implements a storage management API as an integration point for cloud management platforms and developers that need to automate the provisioning and management of the cloud storage system.

Service Proxy

- [Using the Service Proxy](#)
- [How the Service Proxy Works](#)

Service Proxy is a front-end protocol for Content Gateway that enables cluster administration (via the Swarm UI and the management API), giving you a single access point for managing and monitoring your entire Swarm cluster. With the Service Proxy, you can host Swarm cluster administration from a server that is accessible to your admins and have it manage their communication with the cluster.

The Service Proxy protocol is enabled and configured on a Gateway server through the [Gateway's configuration file](#). The Service Proxy provides access to the management API that is built into the Swarm cluster nodes, using the same IDSYS authorization and authentication as your Content Gateway.

Best practices

Access – Enable the Service Proxy for *cluster admins only*, to grant them alone access to the cluster's Swarm UI and Management API. Disable Service Proxy for all other users (end users, tenants, customers), who should be restricted to the content interfaces (Content UI and the SCSP and S3 APIs).

Production – In production, have one Gateway dedicated to run as Service Proxy for your cluster administration (via Swarm UI and Management API), and have a pool of additional Gateways to handle all content management at scale. *Only if* the cluster is for testing or light usage should you enable *both* cluster administration and content management on a single Gateway instance, such as on a CSN.

Using the Service Proxy

To enable users to log in via the Service Proxy, provide them with the correct URL.

- **Host** – Rather than use the IP address or hostname of a Swarm storage cluster node, give the Service Proxy hostname or IP instead. When using a hostname, verify that DNS resolves the name to the front-end IP address of the Gateway instance that is running Service Proxy.
- **Port** – Include the `bindPort` value (from the `[cluster_admin]` section of [Gateway configuration](#)).

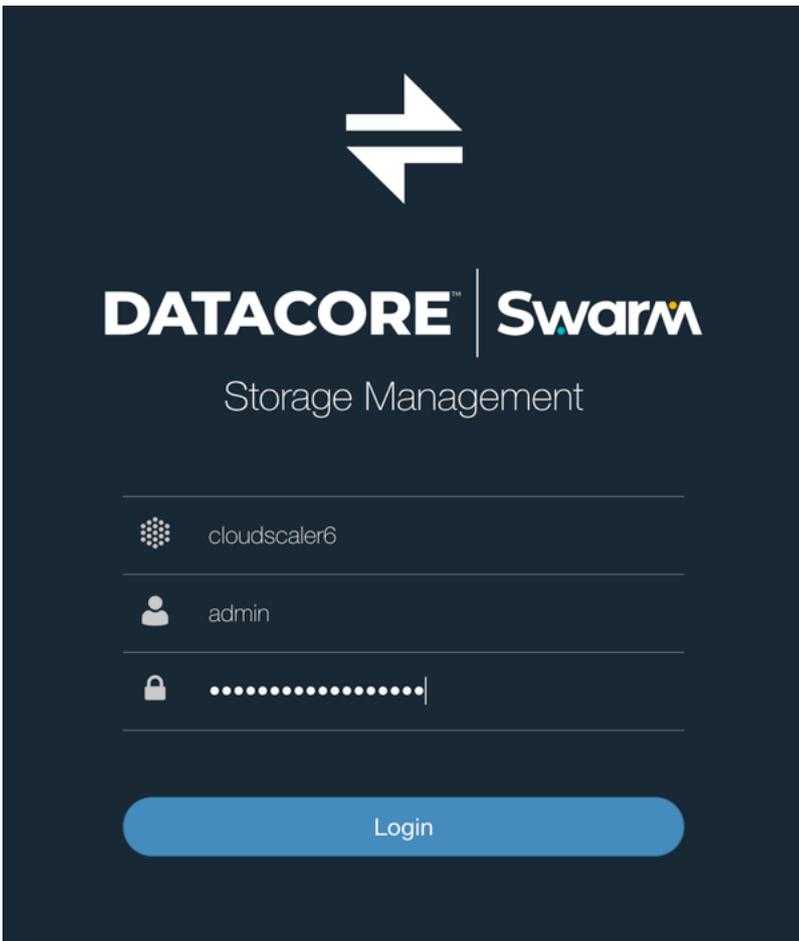
Swarm UI access

`http://HOST:CLUSTER_ADMIN·BINDPORT/_admin/storage`

`http://HOST:91/_admin/storage` (default)

Once you've reached the UI, you will be prompted to log in:

Host	<i>Read-only.</i> The Service Proxy host name or IP address for the Swarm storage cluster to be viewed.
Username	User logins for the UIs are not Swarm-managed but rather LDAP or PAM, as configured by the Gateway IDSYS file, <code>/etc</code>
Password	<code>/caringo/cloudgateway/idsys.json</code> . See Gateway Identity System .



How the Service Proxy Works

The Service Proxy servlet listens on the specified port and handles two types of requests on the same port:

- Storage cluster management API requests, targeting storage nodes
- Elasticsearch query requests, targeting Elasticsearch nodes

Authentication and authorization for the Service Proxy uses Content Gateway's root [IDSYS](#) and root [Policy](#).

See [Gateway Configuration](#) for configuring the Service Proxy and [Content Gateway Authentication](#) for details on authentication/authorization.

Upgrading Gateway



Older Gateway

See [Upgrading from Gateway 5.x](#) if not on Gateway version 6 or higher.

Perform rolling upgrades of Gateway 6.0 and higher:

1. (*Load balancer users only*) In the load balancer, disable traffic for the specific Gateway being upgraded. Allow traffic to continue flowing to the other Gateways.

2. Stop the Gateway service.

```
systemctl stop cloudgateway
```

3. Apply any operating system patches before upgrading the Gateway.

4. Upgrade the Gateway service:

```
yum -y install caringo-gateway-VERSION.rpm
```

5. Reload the systemd control scripts:

```
systemctl daemon-reload
```

6. Upgrade the Swarm Storage UI:

```
yum install caringo-storage-webui-VERSION.rpm
```

7. Upgrade the Content UI:

```
yum -y install caringo-gateway-webui-VERSION.rpm
```

8. Enable and start the Gateway service:

```
systemctl enable cloudgateway
systemctl start cloudgateway
```

9. (*Load balancer users only*) In the load balancer, re-enable client traffic to the newly upgraded Gateway.

10. Repeat the process for the remaining Gateways.

Upgrading from Gateway 5.x

The Content Gateway software components are packaged as RPMs in the Swarm distribution bundles downloaded from the [Downloads section](#) on the [DataCore Support Portal](#). Download the bundles for both Swarm 11 and Swarm 12 to verify all versions needed to step through a migration from an unsupported version of Elasticsearch are available.



Elasticsearch migration

Gateway 6 and higher cannot work with Elasticsearch 2.3.3, so be ready to switch over to the new search index *before* upgrading Gateway. Perform a rolling upgrade to the new Elasticsearch, taking advantage of the fact that each Gateway does not switch to the new Elasticsearch cluster until it is rebooted if more than one Gateway exists.

Work with DataCore Support to plan well and avoid down-time. See [How to Upgrade Swarm](#), [Upgrading from Unsupported Elasticsearch](#).

1. Review the upgrade impacts and known issues for the release being upgraded to. See [Content Gateway Release Notes](#).
2. Complete the upgrade of the Swarm Storage cluster.
3. Complete the migration to Elasticsearch 6, so it is the primary search feed Gateway uses on restarting after upgrade. See [Migrating from Older Elasticsearch](#).
4. (*Load balancer users only*) In the load balancer, disable traffic for the specific Gateway being upgraded. Allow traffic to continue flowing to the other Gateways.
5. Stop the Gateway service.


```
systemctl stop cloudgateway
```
6. Apply any operating system patches before upgrading the Gateway.
7. Upgrade the RPMs for Gateway, Swarm UI, and Content UI:


```
yum -y install caringo-gateway-VERSION.rpm
yum -y install caringo-storage-webui-VERSION.rpm
yum -y install caringo-gateway-webui-VERSION.rpm
```
8. Reload the systemd control scripts:


```
systemctl daemon-reload
```
9. Fall back and retry if the upgrade failed:
 - a. Revert the RPM back to Gateway 5.x.
 - b. Restart the Gateway that failed the upgrade.
10. Review and modify the `gateway.cfg` and the new `logging.yaml` configuration files in the `/etc/caringo/cloudgateway` directory. See [Gateway Configuration](#).
 - `gateway.cfg` – In the **[storage_cluster]** section, update the `indexerHosts` value in to the new Elasticsearch server.
 - `logging.yaml` – Edit the **Syslog** block and change the protocol to TCP if the server supports TCP and it is desired.
11. The Gateway service does not automatically start after a system reboot, so re-enable the service if upgrading from version 5.2.1 or earlier. (CLOUD-2819)


```
systemctl enable cloudgateway
```
12. Start the Gateway service:


```
systemctl start cloudgateway
```
13. (*Load balancer users only*) Re-enable client traffic to the newly upgraded Gateway in the load balancer.
14. Repeat the process for the remaining Gateways.

Content Gateway Authentication

- [Gateway Identity System](#)
- [Gateway Access Control Policies](#)

Gateway Identity System

The Content Gateway identity system is implemented by the front-end Gateway component and allows for one or more user data store configurations. The user data store is responsible for authenticating users via a password and for defining group membership for users. Content Gateway currently supports LDAP and Linux PAM as the user data store. Microsoft Active Directory can be used through its LDAP interface or via PAM with a Kerberos configuration on the Gateway server.

SAML

As of Swarm 12 and Content Gateway 7.1, you can also enable single sign-on to both Swarm UI and Content UI through your third-party Identity Provider. See [Enabling SSO with SAML](#). (v7.1)

The configuration of the identity system exists as IDSYS documents that are stored in the following locations:

- Root IDSYS file
- Tenant IDSYS sub-resource
- Storage domain IDSYS sub-resource

An IDSYS document contains the information necessary to connect with the identity system and defines the organization of users and groups within the identity system. While an IDSYS document may only define one back-end identity system, different back-end systems can be used for different tenants and storage domains within the cluster. For example: use PAM in the root IDSYS and use LDAP in the storage domains.

The root IDSYS is stored in this JSON file:

```
/etc/caringo/cloudgateway/idsys.json
```

This file *must* be kept synchronized between all Gateway servers. The `idsys` sub-resource for a tenant or storage domain is kept within the cluster, is shared among all Gateway servers, and is accessed through the Gateway Management API or the Storage API.

Note

The root IDSYS configuration file must exist and must contain a valid JSON string or be blank. The minimum valid JSON content is "{}".

- [IDSYS Document Format](#)
- [Enabling SSO with SAML](#)

IDSYS Document Format

IDSYS documents are JSON-formatted objects and are specific to the back-end identity management system: Active Directory, LDAP, and Linux PAM.

SAML for SSO

With Gateway 7.1, Content UI 7.0, and Swarm UI 3.0 and higher, can enable SSO (single sign-on) to the Swarm and Content UIs through a third-party identity provider, such as Google. See [Enabling SSO with SAML](#). (v7.1)

- [SAML for SSO](#)
- [Common IDSYS Fields](#)
 - [Password security](#)
- [LDAP and AD Examples](#)
 - [LDAP](#)
 - [Active Directory](#)
- [LDAP and AD Fields](#)
 - [No nested or recursive groups](#)
 - [Important](#)
- [PAM Example](#)
- [Modifying IDSYS](#)
 - [Important](#)
 - [Caution](#)
- [IDSYS Precedence Model](#)
- [Qualifying User and Group Names](#)
 - [Best practices](#)

Common IDSYS Fields

Below are the common fields within all IDSYS documents. Fields specific to the back-end identity management system are broken out into separate sections.

Field	Required	Description
name	No	Name of the IDSYS document; value is not used by Gateway
description	No	Description of the IDSYS document; value is not used by Gateway
comments	No	Comments about the IDSYS; may be any valid JSON object type
cookieName	No ¹	Cookie name used to store authentication token. Example: "token"
tokenPath	No ¹	URI path used for token authentication. Example: "/.TOKEN/"
tokenAdmin	No ¹	User allowed to view and delete authentication tokens for other users.

¹ For details regarding token-based authentication, see [Token-Based Authentication](#).

When a user authenticates to the Gateway using HTTP Basic authentication (not token-based authentication and not S3 HMAC), the user's password is stored in the normal field for LDAP or PAM and it may be hashed in whatever formats are supported by the system. For LDAP, this field is normally `userPassword`; for PAM with the traditional Unix authentication mechanism, it is the second field in the `/etc/shadow` file.



Password security

Plain-text passwords in both [Gateway Configuration](#) and IDSYS are replaced by encrypted versions on startup. Enter the new credentials and restart Gateway, when changing management passwords, which replaces those strings with encrypted versions as part of the startup. (v7.1)

LDAP and AD Examples

These are examples of IDSYS documents for LDAP and Active Directory. They contain fields specific to LDAP as well as fields common to all IDSYS documents.

LDAP

```
{ "ldap": {
  "name" : "idsys-ldap",
  "description": "LDAP identity management configuration",

  "protocol" : "ldaps",
  "ldaphost": [ "ldap.example.com", "ldap-sec.example.com" ],
  "ldapport": 636,

  "adminDN": "uid=USERNAME,ou=Users,dc=example,dc=com",
  "adminPassword": "PASSWORD",

  "userBase": "ou=Users,dc=example,dc=com",
  "groupBase": "ou=Groups,dc=example,dc=com",

  "userFilter": "objectclass=account",
  "groupMemberUidAttr": "memberUid",

  "cookieName": "token",
  "tokenPath": "/.TOKEN/",
  "tokenAdmin": "superuser@admindomain.example.com"
} }
```

The block beginning with "uidAttribute" makes this specific to Active Directory:

Active Directory

```
{ "ldap": {
  "name": "idsys-ad",
  "description": "Active Directory example configuration",

  "protocol" : "ldaps",
  "ldaphost": "ad.mycompany.com",
  "ldapport": "636",

  "adminDN": "cn=BINDUSER,ou=Applications,dc=mycompany,dc=com",
  "adminPassword": "BINDPASSWORD",

  "userBase": "ou=Users,dc=mycompany,dc=com",
  "groupBase": "ou=Groups,dc=mycompany,dc=com",

  "uidAttribute": "sAMAccountName",
  "userFilter": "objectclass=",
  "groupMemberDNAttr": "member",

  "cookieName": "token",
  "tokenPath": "/.TOKEN/",
  "tokenAdmin": "caringoadmin@"
}
```

LDAP and AD Fields

These are the fields within the IDSYS document specific to the LDAP or Active Directory back-end identity management system.



No nested or recursive groups

Nested/recursive groups, such as the built-in groups in Active Directory, are not supported by Gateway.

Field	Default	Required	Description
ldaphost		Yes	Host name or IP of the LDAP server as a string or a multiple servers as a list. Example: ["1.1.1.1", "1.1.1.2"]
ldapport		Yes	Port the LDAP service is running on
protocol	ldap	No	Set to "ldap" or "ldaps"
referrals	follow	No	Set to "follow" or "ignore" to control how referrals are handled
adminDN		Yes	DN used to bind to the LDAP server for queries
adminPassword		Yes	Password for adminDN user
userBase		Yes	DN where users are defined
groupBase		Yes	DN where groups are defined
uidAttribute	uid	No	Attribute name containing user's ID. Examples: <ul style="list-style-type: none"> "uid" for OpenLDAP and ApacheDS "sAMAccountName" for Active Directory
userFilter		Yes	Filter for user objects. Example: "objectclass=account"

groupMemberUidAttr		Yes ¹	Group attribute whose values contain uid of member. Example: "memberUid" if OpenLDAP is configured for groups with "objectclass=posixgroup"
groupMemberDNAttr		Yes ¹	Group attribute whose values contain DN of member. Example: "member" if OpenLDAP is configured for groups with "objectclass=groupOfNames"; also common with Active Directory
s3SecretKeyAttr		No ²	**Deprecated** User attribute whose value contains the user's S3 secret key in plain-text. Verify "userPassword" has a plaintext value since this is not the normal handling of this attribute if used.

¹ The `groupMemberUidAttr` and `groupMemberDNAttr` parameters are mutually exclusive and one must be defined in IDSYS.

² The `s3SecretKeyAttr` parameter is needed when using S3 Protocol Personality with a user password stored in LDAP. It is not required when using token authentication exclusively.

The `adminDN` and `adminPassword` parameters define the credentials with which the Gateway binds to the LDAP system to perform queries and read records for users and groups. The `adminDN` entity within LDAP needs to have read level access (`rscdx` privileges) within the LDAP tree. It is not necessary to grant write or manage level access to Gateway.

- A user's name in an access control Policy document is the value of the LDAP attribute named by the `uidAttribute` parameter. This is the `uid` attribute of a user's LDAP record by default .
- A group's name in an access control Policy document is the `cn` attribute for the group LDAP entity. The name of this attribute cannot be configured. A group's name may contain spaces and other non-alphanumeric characters.



Important

Add the signer's public key PEM file to the Java keystore to avoid a `SunCertPathBuilderException` when Gateway queries the LDAP server if using LDAPs with self-signed certificates.

Although previously this required running `java keytool`, now use the CentOS/RHEL utility `update-ca-trust` to add any CA to the system.

```
# cp ca.pem /etc/pki/ca-trust/source/anchors/
# update-ca-trust extract
```

PAM Example

There are no fields within the IDSYS document specific to the PAM back-end identity management system. Follow this process to implement identity management if using PAM:

1. Because the `root` user (`uid=0`) on this Content Gateway server cannot be used to authenticate to the Gateway, create another user (such as `superuser@admindomain.example.com`) on this server for this purpose.
2. Copy and paste this example into the IDSYS document: `/etc/caringo/cloudscaler/idsys.json`.

```
{ "pam": {
  "name" : "idsys-pam",
  "description": "PAM identity management configuration",
  "cookieName": "token",
  "tokenPath": "/.TOKEN/",
  "tokenAdmin": "superuser@admindomain.example.com"
} }
```

3. Update the `tokenAdmin` to match the authentication user.

Modifying IDSYS

The root IDSYS configuration is stored in the `idsys.json` file on the Gateway server's disk so it is always available and an administrator can always modify it. This prevents locking oneself out of the storage cluster. Changes to the local file take effect without the need to restart the Gateway.

Important

It is crucial the root IDSYS document is synchronized across all servers when more than one Gateway server is deployed.

See [Defined ETC Documents](#) for modifying a tenant or storage domain's sub-resource through the management API.

See [SCSP Context Sub-resources](#) for details on modifying a storage domain's sub-resource through the storage API.

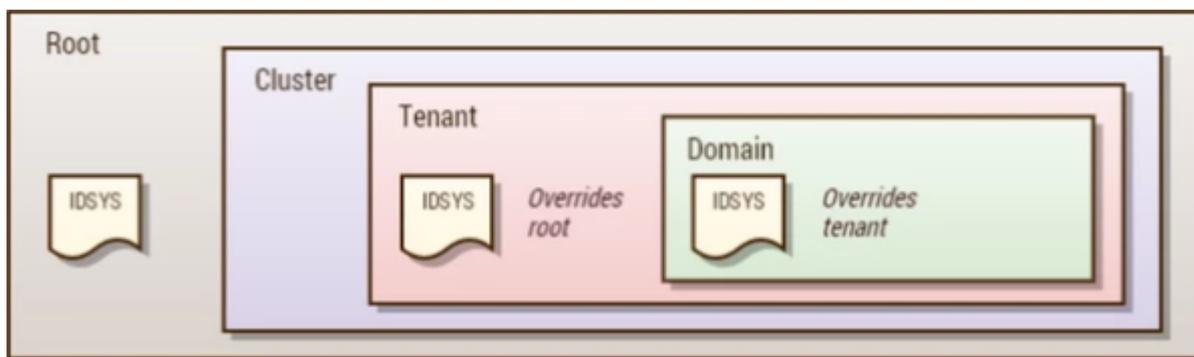
The entire JSON document with all fields must be provided in the update request, even if one field is being modified when updating an IDSYS sub-resource through the management API or the storage API.

Caution

Protect permission to read and update the IDSYS document for a domain from untrusted users. In deployments where a service provider allows clients to manage content within domains, the service provider normally maintains sole privilege to access the IDSYS document. This includes retaining ownership of the domain objects.

IDSYS Precedence Model

The identity system is described by IDSYS documents can exist at the root, tenant, and storage domain within the system. The lowest level overrides the higher levels when IDSYS documents exist at multiple levels in the hierarchy. It inherits from a higher level when a lower level lacks an IDSYS.



all tenants and all storage domains inherit from the Root IDSYS if a Root IDSYS exists. There is one identity management system with one set of users and groups. Each tenant and the

storage domains owned by them share an identity system separate from the Root IDSYS if the tenants each defines an IDSYS. The storage domains inherit from the Tenant IDSYS.

The IDSYS inheritance also works at the field level. Tenant and storage domain IDSYS documents can choose to override specific fields. The value is inherited by the tenant and domain levels if `tokenAdmin` is defined in the Root IDSYS and not in the tenant or domain IDSYS. The Root IDSYS may define the LDAP `adminDN` and `adminPassword` and allows the tenant and domain IDSYS documents to override the `userBase` and `groupBase` values.

- **Single Company**
 - In this scenario, the company has one identity management system, there is one tenant per business unit, and each business unit has one or more storage domains. This scenario is likely with a private cloud serving a single company. The configuration in this scenario is the Root IDSYS defining the configuration of the identity management system and there are no IDSYS definitions for the tenants and storage domains. Therefore, the tenants and storage domains inherit from the Root IDSYS using a single source of users and groups.
- **Service Provider / Distributed Company**
 - In this scenario, a storage MSP, or a large company with business units each with separate identity management systems and multiple user/group sources. The configuration in this scenario is the Root IDSYS defining the cluster administrator users and groups and the Tenant IDSYS documents defining the users and groups for each client or business unit. The storage domains do not define an IDSYS so they inherit the definition from the tenant and share the users and groups with the other storage domains owned by the tenant.
- **Service Provider with Resellers**
 - This is an extension of the previous scenario except each tenant can be a reseller offering storage domains to separate, unrelated companies. In this case, each storage domain defined an IDSYS that overrides the Tenant IDSYS allowing a different set of users and groups for each storage domain. This scenario is not mutually exclusive with the previous one: a hybrid of the two is possible where some domains override the IDSYS of the tenant, and others do not.

Qualifying User and Group Names

It may be required to fully qualify the user and group principal names to verify correct policy resolution. In access control policies and `x-owner-meta` headers, a "fully qualified" principal has a tenant name or storage domain appended directly to the name:

<code>user</code> <code>group</code>	non-qualified	Principal from the same IDSYS scope as the content
<code>user@domain</code> <code>group@domain</code>	fully qualified	Principal from a specific storage domain's IDSYS scope
<code>user+tenant</code> <code>group+tenant</code>	fully qualified	Principal from a specific tenant's IDSYS scope
<code>user@</code> <code>group@</code>	fully qualified	Principal from root scope

The name in the policies may remain unqualified (no `@domain` or `+tenant` suffix on principal names) if a principal (user/group) authenticates from the *same* IDSYS as the resource they are accessing.

Gateway uses the default assignment of the `x-owner-meta` header value to fully qualify the principal (such as `user@domain` or `user+tenant`) if a principal authenticates from a *different* IDSYS from the one used by the resource. Applications can also assign object ownership across domains, where the IDSYS of the storage domain differs from the user from another domain. There is no limit on the number of cross-domain relationships that exist, but all must be within the same Swarm cluster.

Tokens – Tokens are bound to the IDSYS of the context both *where and when* they are created. The token has to take the root scope if creating a tenant-level token but the tenant does not have an IDSYS. All requests using this token authenticate using the root IDSYS (and likely fail, not finding the user there), even if a correct tenant-level IDSYS is added later. The token must ignore any domain-level IDSYS, current or future if creating a tenant-level token with a tenant IDSYS. Either create a tenant-level IDSYS and use `inherit` at the domain-level or create tokens at the domain-level if domain-level controls over tokens is desired.

 **Best practices**

Fully qualify any **token administrators** defined in an IDSYS document. This practice avoids ambiguity if a storage domain inherits an IDSYS from the tenant or root scope because token administrator is a privileged permission.

Fully qualify user/group names in the policies if there is **more than one IDSYS** involved, to verify there are no problems with policy resolution.

Enabling SSO with SAML

By implementing SAML, users can log in to Swarm browser components (Swarm UI and Content UI) using the exiting credentials from another source – which is known as *single sign-on* (SSO). SAML (Security Assertion Markup Language) is an open standard that allows *identity providers* (IdP) pass along authorization credentials to *service providers* (SP). The popularity stems from the all-around benefits:

- **User friendly** – SAML simplifies life for users with password elimination, fast logins, and automatically renewing sessions.
- **IT friendly** – SAML streamlines life for IT with centralized authentication, strong digital signatures, and easy directory integration.

With SAML integration, Content Gateway becomes a Service Provider in the authentication flow, interacting with the IdP to authenticate users. (v7.1)



Version requirements

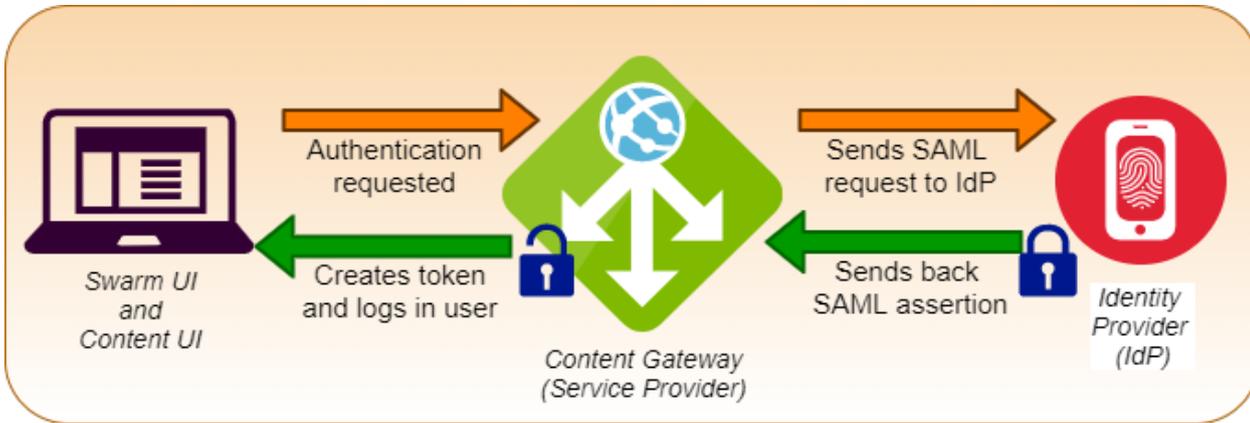
For SAML authentication support, upgrade to these or newer versions: Content Gateway 7.1, Content UI 7.0, and Swarm UI 3.0.

- [Version requirements](#)
- [How SAML Works in Gateway](#)
 - [Verified Identity Providers](#)
 - [When Tokens Expire](#)
- [Implementing SAML](#)
 - [Best practice](#)
 - [IDSYS SAML Fields](#)
 - [SAML Groups for ACL Policies](#)
 - [Group support](#)
- [Troubleshooting SAML](#)
 - [Global SAML properties](#)
 - [Important](#)

How SAML Works in Gateway

While the SAML 2.0 single sign-on (SSO) standard is broad and varied, this section covers only the aspects needed to achieve authentication with Gateway.

Gateway allows use of trusted third-party IdPs to authenticate interactive users for *browser use*. This is the process:



1. When a user accesses Swarm UI or Content UI in an unauthenticated browser session, Gateway redirects them to the login page for the identity provider (IdP).
2. The user authenticates with the IdP.
3. The IdP redirects back to Gateway with digitally signed proof (the *assertion*) of successful authentication on success.
4. Gateway decrypts the assertion, extracts user and group information, and generates a Swarm token.
5. Gateway redirects back to the original URL with a *cookie* that carries the token UUID for the rest of the browser session.

Swarm supports these SAML 2.0 bindings:

- From Gateway to IdP: both SSO (sign-on) and SLO (log-out) via `HTTP-Redirect`
- From IdP to Gateway: SSO assertions via `HTTP-POST`, SLO responses via `HTTP-Redirect`

Only these interactions are supported:

- Signed and encrypted assertions from the IdP
- Gateway-initiated requests

Verified Identity Providers

Gateway's SAML support has been verified with these leading IdPs. Following are features and differences found among them.

IdP	Features	Limitations
OneLogin	<ul style="list-style-type: none"> • Supports Single Sign-On (SSO) • Supports Single Logout (SLO) • Supports encrypted Assertions via public/private keypair (). • Supports groups via custom attributes 	
Okta	<ul style="list-style-type: none"> • Supports Single Sign-On (SSO) • Supports groups via custom attributes 	<ul style="list-style-type: none"> • Partial support for Single Logout (SLO) • No public/private keypair support
Google	<ul style="list-style-type: none"> • Supports Single Sign-On (SSO) • Requires HTTPS on redirect to Gateway • Supports groups via custom attributes 	<ul style="list-style-type: none"> • No support for Single Logout (SLO) • No public/private keypair support

Configure the IDSYS with an empty URL for SLO if the IdP has incomplete Single Logout (SLO) support. Gateway deletes the token without contacting the IdP upon logout. A subsequent login may not require the user to enter credentials.

When Tokens Expire

Gateway creates a token with an expiring specified by the IdP in the Assertion if authentication with the IdP succeeds. Usually this defaults to 1440 minutes, which is one day.

Gateway falls back to the configured expiration, which is the `[gateway] tokenTTLHours` if it cannot determine when the session expires from the Assertion. This expiration defaults to one day.

See [Gateway Configuration](#).

Implementing SAML

As with other types of Gateway authentication (such as PAM and LDAP), a root IDSYS must exist but additional ones can be configured. Add an IDSYS for a specific domain or tenant if single sign-on is desired. See [Gateway Identity System](#).



Best practice

Choose a strategy to guarantee super users can always gain access in case the SAML method is misconfigured or goes offline if implementing global SSO. These are two approaches:

- **Company-wide tenant for SSO** – Assign super users to root-level PAM:
 1. *Root IDSYS*: Using Linux PAM, define super users as Linux users on the Gateway servers.
 2. *Tenant IDSYS*: Using SAML, enable SSO for *all* users under an organization-wide Content UI tenant (see step 4).
 3. *Swarm UI access*: Grant storage admin access to additional users as needed from the root IDSYS.
- **Special domain for super users** – Assign super users to a domain-level PAM, and have them browse to the DNS host name that matches the storage domain where the PAM IDSYS resides:
 1. *Root IDSYS*: Using SAML, enable SSO for *all* users via the root IDSYS (see step 3).
 2. *Domain IDSYS*: Using Linux PAM, define super users as Linux users the Gateway servers.
 3. *Swarm UI access*: Grant storage admin access to additional users as needed from the special domain.

The Gateway configuration and the setup required by the IdP need to be completed to enable SAML login capability. None of the Gateway-side configuration requires restarting of Gateway services.

1. In the administrative portal for the IdP, start setting up SAML to obtain the URLs and public/private keys needed for configuring Gateway.
2. **Root SSO**, if applicable:
 - a. Edit the root IDSYS and configure it for SAML. See *IDSYS SAML Fields*, below, and [IDSYS Document Format](#).
 - b. Edit the root Policy and verify it grants the permissions needed.
See *SAML Groups for ACL Policies*, below, and help on the [Policy Document](#).
3. **Tenant/Domain SSO**, if applicable: To have separate SSO for specific tenants or domains, complete that setup in the Content UI. For each tenant or domain,
 - a. Navigate to **Properties** (gear icon), and scroll to **Identity Management**.
See [Setting Identity Management](#).

- b. Uncheck **Inherited** and from the **Templates** drop-list select **SAML**.
 - c. Configure the SAML. See *IDSYS SAML Fields*, below.
 - d. Obtain the attributes needed to complete the remainder of the setup with the identity provider.
 - i. Below the script area, under **Identity Provider (IdP) Resources**, click the links.
 - ii. Open, copy, and store the **Service Provider Attributes**, which include the endpoints needed for assertions and logouts.
 - iii. Download the XML file version to the right and use that if that service provider can import SAML metadata.
 - e. Open the **Permissions** tab and verify it grants the permissions needed.
See *SAML Groups for ACL Policies*, below, and [Setting Permissions](#).
4. Back in the administrative portal for IdP, complete any remaining setup using the values and endpoints from Gateway.
 5. Verify single sign-on is working as expected:
 - a. When a user browses to a storage domain, they see the normal login page *unless* that domain has a SAML IDSYS (inherited or explicit), which triggers the SAML login process.
Important: Gateway uses the IDSYS of the storage domain that matches the DNS host name in the URL.
 - b. If the name they type includes a context suffix (`jdoe+tenant` or `jdoe@domain`), the applicable IDSYS is checked.
 - c. If a SAML login is triggered, the password field is disabled and the **Login with SSO** button takes them to the identity provider to complete login.



IDSYS SAML Fields

This is the streamlined template filled out to create an IDSYS for SAML in the Content UI, to apply to a tenant or domain. It has fewer fields because the Content UI generates the set of Service Provider ("sp*") fields automatically:

```

{
  "saml": {
    "cookieName": "token",
    "tokenPath": "/.TOKEN/",
    "entity": "Your root organization",
    "idpEntityId": "The Identity Prov.",
    "idpSsoUrl": "The Identity Provid.",
    "idpSloUrl": "The Identity Provid.",
    "groupAttrName": "User attribute :",
    "idpCert": ""
  }
}

```

}

This is a sample IDSYS.json configuration for authenticating via [OneLogin](#), which shows the Service Provider ("sp*") fields:

```
{
"saml": {
  "cookieName": "token",
  "tokenPath": "/.TOKEN/",
  "spEntityId": "https://atomic.com/caringo",
  "spAcsUrl": "http://gw.atomic.com:8888/_admin/saml/login",
  "spSloUrl": "http://gw.atomic.com:8888/_admin/saml/logout",
  "idpEntityId": "https://app.onelogin.com/saml/metadata/9f23...b1cda",
  "idpSsoUrl": "https://caringo.onelogin.com/trust/saml2/http-redirect/sso/9f2...cda",
  "idpSlorl": "https://caringo.onelogin.com/trust/saml2/http-redirect/slo/125...502",
  "groupAttrName": "group1, group2",
  "idpCert": "MIID3DCC...N8U8nVLp7Ka0="
  "spPrivateKey": "---BEGIN PRIVATE KEY---\nMII...Yts=\n---END PRIVATE KEY---\n"
}
}
```

This is how these fields are used and how they differ between a root IDSYS and one created through the Content UI:

All IDSYS Fields	Content UI Template	Source	
cookieName	cookieName		The name of the cookie that carrying the token UUID.
tokenPath	tokenPath		The URL used to create the Swarm token.
spEntityId	entity <i>(used in final generated value)</i>		For the IdP, identifies this particular SAML IDSYS instance. This must be a unique and valid URL. Each <i>must</i> be unique if multiple tenants and domains exist and Gateway is configured with multiple SAML IDSYS instances. <i>Tip:</i> The Content UI generates the final spEntityId value and guarantees uniqueness for the user. A single "entity" value may be reused across tenants and domains.
spAcsUrl	<i>(generated)</i>		After a successful login with the IdP, this is the redirect target, the URL where Gateway receives the Assertion. For any IDSYS other than the root, the URL path must include <i>/tenant/domain</i> . This be communicates to the IdP.
spSloUrl	<i>(generated)</i>		After a successful logout from the IdP, this is the URL where Gateway expects a callback. For any IDSYS other than the root, the URL path must include <i>/tenant/domain</i> . This communicates to the IdP.
idpEntityId	idpEntityId	IdP	The unique ID of the IdP. This URL comes from the IdP during the administrative setup phase.
idpSsoUrl	idpSsoUrl	IdP	Where Gateway redirects to initiate a login at the IdP. This information comes from the IdP during the administrative setup phase. Always choose REDIRECT if the IdP offers both POST and REDIRECT login options.

idpSloUrl	idpSloUrl	IdP	<p>Where Gateway redirects to initiate a logout at the IdP. This information comes from the IdP during the administrative setup phase.</p> <p>Always choose <code>REDIRECT</code> if the IdP offers both <code>POST</code> and <code>REDIRECT</code> logout options.</p> <p>Leave this blank if the IdP does not support Single Logout (SLO). Gateway deletes the token without contacting the IdP upon logout.</p>
idpCert	idpCert	IdP	<p>The IdP's X.509 public key certificate, encoded as Privacy-Enhanced Mail (PEM). This information comes from the IdP during the administrative setup phase.</p>
wantAssertionsEncrypted			<p>Whether to encrypt the Assertion sent <i>from</i> the IdP. Enabling encryption requires certificates of public/private key configuration.</p> <p>Logins get this error if this is enabled but both parts of the key pair are not configure :</p> <pre>500 Errors detected in SAML settings [idp_cert_or_fingerprint_not_found_and_required]</pre>
spPrivateKey		IdP	<p><i>Required when encrypting Assertions.</i> Set this to the private key for <i>this</i> SAML IDSYS instance.</p> <p>The IdP must be configured with the matching public key used to encrypt Assertions. The Gateway decrypts using this private key, implementing a trust relationship without the complexity of certificates.</p> <p>Both keys must be configured during the administrative setup phase. Keys must be in Public-Key Cryptography Standards (PKCS) #8 format.</p>
groupAttrName	groupAttrName	IdP	<p>(Optional) List of one or more names to be treated as user group attributes, based on groups maintained by the IdP.</p>
groupSeparator		IdP	<p>Character that separates multiple group attribute names in the list. Defaults to comma.</p>

These fields are also supported in the IDSYS JSON config:

- spCert
- nameIdEncrypted
- authRequestSigned
- logoutRequestSigned
- logoutResponseSigned
- signMetadata
- wantMessagesSigned
- wantAssertionsSigned
- wantNameIdEncrypted

SAML Groups for ACL Policies

To avoid having to name every user, Gateway authorization policies typically refer to groups. To skip manual maintenance of group membership, Gateway automatically assigns the SAML users to groups referenced in policies, and it also allows importing groups. These are the types of groups users can belong to:

Scope	Group Name	Group Examples	About this Group
Global	SAMLUsers		Every user authenticated via SAML joins this default group.
Domain	<domain>	gmail.com	Every user who shares the same @<domain> joins a group named for <i>that</i> domain. jdoe@gmail.com belongs to group gmail.com .
Attribute (optional)	<attribute> <attribute>.<domain>	dev admin dev.gmail.com admin.gmail.com	Add attribute names to the Assertions to reference existing groups. Set the field <code>groupAttrNames</code> to a comma-separated list of one or more names. When an Assertion includes attribute names, each name listed becomes a group that the user joins. For any group names in the form <code>group@domain</code> , Gateway avoids syntax conflict with Swarm domains by replacing '@' with a period: <code>group.domain</code> .



Group support

All major IdPs discussed here (OneLogin, Okta, and Google) support groups via these custom attribute names.

Troubleshooting SAML

Edit the configuration to enable SAML-specific debugging:

1. Edit the global SAML properties file on the Gateway server.
2. Enable debug mode:

```
onelogin.saml2.debug = true
```

No restart of the Gateway service is required; settings are reloaded for each login request.

3. Repeat these steps to disable debug level once debugging is finished.

Global SAML properties

Gateway installs with a global SAML properties file that uses the open source toolkit from OneLogin, and it supports deployments to *all* IdPs. There is no need to use it except for troubleshooting with DataCore Support.



Important

Those settings prefixed by [CONFIG] are *overridden* by the values in a given IDSYS, if they differ.

This is a version of the properties file with the comments trimmed out, so the settings it manages can be focused on :

Properties, trimmed

```

onelogin.saml2.strict = true
onelogin.saml2.debug = false

#####
# Service Provider data for Gateway
#####

[CONFIG] onelogin.saml2.sp.entityid =
[CONFIG] onelogin.saml2.sp.assertion_consumer_service.url =
onelogin.saml2.sp.assertion_consumer_service.binding = urn:oasis:names:tc:SAML:2.0:bindings:HTTP-
[CONFIG] onelogin.saml2.sp.single_logout_service.url =
onelogin.saml2.sp.single_logout_service.binding = urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redir
onelogin.saml2.sp.nameidformat = urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified
[CONFIG] onelogin.saml2.sp.x509cert =
[CONFIG] onelogin.saml2.sp.privatekey =

#####
# Identity Provider data to connect with Gateway
#####

[CONFIG] onelogin.saml2.idp.entityid =
[CONFIG] onelogin.saml2.idp.single_sign_on_service.url =
onelogin.saml2.idp.single_sign_on_service.binding = urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Red
[CONFIG] onelogin.saml2.idp.single_logout_service.url =
[CONFIG] onelogin.saml2.idp.single_logout_service.response.url =
onelogin.saml2.idp.single_logout_service.binding = urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redi
[CONFIG] onelogin.saml2.idp.x509cert =

#####
# Security settings
#####

[CONFIG] onelogin.saml2.security.nameid_encrypted = false
[CONFIG] onelogin.saml2.security.authnrequest_signed = false
[CONFIG] onelogin.saml2.security.logoutrequest_signed = false
[CONFIG] onelogin.saml2.security.logoutresponse_signed = false
[CONFIG] onelogin.saml2.security.want_messages_signed = false
[CONFIG] onelogin.saml2.security.want_assertions_signed = false
[CONFIG] onelogin.saml2.security.sign_metadata =
[CONFIG] onelogin.saml2.security.want_assertions_encrypted = false
[CONFIG] onelogin.saml2.security.want_nameid_encrypted = false
onelogin.saml2.security.requested_authncontext = urn:oasis:names:tc:SAML:2.0:ac:classes:Password
onelogin.saml2.security.onelogin.saml2.security.requested_authncontextcomparison = exact
onelogin.saml2.security.want_xml_validation = true
onelogin.saml2.security.signature_algorithm = http://www.w3.org/2000/09/xmldsig#rsa-sha1

onelogin.saml2.organization.name = Caringo
onelogin.saml2.organization.displayname = Caringo, Inc.
onelogin.saml2.organization.url = http://www.caringo.com
onelogin.saml2.organization.lang =

onelogin.saml2.contacts.technical.given_name = Caringo Support
onelogin.saml2.contacts.technical.email_address = support@caringo.com
onelogin.saml2.contacts.support.given_name = Caringo Support
onelogin.saml2.contacts.support.email_address = support@caringo.com

```

<p>onelogin.saml2.strict = true</p>	<p>These are rejected if enabled:</p> <ul style="list-style-type: none"> • Unsigned or unencrypted messages that are signed or encrypted • Messages not strictly following the SAML 2 standard
-------------------------------------	--

onelogin.saml2.debug = false	Debug mode allows error printing if enabled. Use only during troubleshooting.
SP data	Service Provider Data being deployed
[CONFIG] onelogin.saml2.sp.entityid =	Identifier of the SP (service provider) entity, which must be a URI.
[CONFIG] onelogin.saml2.sp.assertion_consumer_service.url =	Specifies info about where and how the <code><AuthnResponse></code> message must be returned to the requester. This is the URL location where the <code><Response></code> from the IdP return to Gateway.
onelogin.saml2.sp.assertion_consumer_service.binding = urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST	Specifies which SAML protocol binding to use when returning the <code><Response></code> message. For this endpoint, the Toolkit only supports the <code>HTTP-POST</code> binding.
[CONFIG] onelogin.saml2.sp.single_logout_service.url =	Specifies where to return the <code><Logout Response></code> message to the requester, Gateway.
onelogin.saml2.sp.single_logout_service.binding = urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect	Specifies which SAML protocol binding to use when returning the <code><LogoutResponse></code> or sending the <code><LogoutRequest></code> message. For this endpoint, the Toolkit only supports the <code>HTTP-Redirect</code> binding.
onelogin.saml2.sp.nameidformat = urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified	Specifies constraints on the name identifier to be used to represent the requested subject. Gateway supports "unspecified". An email address is required for SSO login. Other formats are not supported.
[CONFIG] onelogin.saml2.sp.x509cert = [CONFIG] onelogin.saml2.sp.privatekey =	Usually the X.509 certificate and the private key of the SP are provided by files placed at the <code>certs</code> folder, but they can be included directly using these parameters. The private key requires <code>Format PKCS#8 BEGIN PRIVATE KEY</code> . Convert it with this command if <code>PKCS#1 BEGIN RSA PRIVATE KEY</code> exists: <pre>openssl pkcs8 -topk8 -inform pem -nocrypt -in sp.rsa_key -outform pem -out sp.pem</pre>
IdP data	Identity Provider Data to connect with our SP
[CONFIG] onelogin.saml2.idp.entityid =	Identifier of the IdP entity (must be a URI)
[CONFIG] onelogin.saml2.idp.single_sign_on_service.url =	SSO endpoint info of the IdP. (Authentication Request protocol) URL Target of the IdP where the SP sends the Authentication Request Message
onelogin.saml2.idp.single_sign_on_service.binding = urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect	SAML protocol binding to be used when returning the <code><Response></code> message. For this endpoint, the Toolkit only supports the <code>HTTP-Redirect</code> binding.
[CONFIG] onelogin.saml2.idp.single_logout_service.url =	SLO endpoint info of the IdP. URL Location of the IdP where the SP sends the SLO Request

<p>[CONFIG] onelogin.saml2.idp. single_logout_service.response.url =</p>	<p>(optional) SLO Response endpoint info of the IdP. URL Location of the IdP where the SP sends the SLO Response.</p> <p>The value for <code>onelogin.saml2.idp.single_logout_service.url</code> is used if left blank. Some IdPs use a separate URL for sending a logout request and response; use this property to set the separate response URL.</p>
<p>onelogin.saml2.idp. single_logout_service.binding = urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect</p>	<p>SAML protocol binding to be used when returning the <Response> message.</p> <p>For this endpoint, the Toolkit only supports the <code>HTTP-Redirect</code> binding.</p>
<p>[CONFIG] onelogin.saml2.idp.x509cert =</p>	<p>Public X.509 certificate of the IdP.</p>
<p>onelogin.saml2.idp.certfingerprint = onelogin.saml2.idp. certfingerprint_algorithm = sha1</p>	<p>Use a fingerprint rather than use the whole X.509 certificate. Generate it with a command (<code>openssl x509 -noout -fingerprint -in "idp.crt"</code>), or add the <code>-sha256</code>, <code>-sha384</code> or <code>-sha512</code> parameter.</p> <p>The <code>certFingerprintAlgorithm</code> is required to allow the toolkit to know which Algorithm was used if a fingerprint is provided. Possible values:</p> <ul style="list-style-type: none"> • sha1 (<i>default</i>) • sha256 • sha384 • sha512
<p>Security settings</p>	
<p>[CONFIG] onelogin.saml2.security. nameid_encrypted = false</p>	<p>Whether the <code>nameID</code> of the <samlp:logoutRequest> sent by this SP is encrypted.</p>
<p>[CONFIG] onelogin.saml2.security. authnrequest_signed = false</p>	<p>Whether the <samlp:AuthnRequest> messages sent by this SP are signed. The Metadata of the SP offers this information.</p>
<p>[CONFIG] onelogin.saml2.security. logoutrequest_signed = false</p>	<p>Whether the <samlp:logoutRequest> messages sent by this SP are signed.</p>
<p>[CONFIG] onelogin.saml2.security. logoutresponse_signed = false</p>	<p>Whether the <samlp:logoutResponse> messages sent by this SP are signed.</p>
<p>[CONFIG] onelogin.saml2.security. want_messages_signed = false</p>	<p>Whether the <samlp:Response>, <samlp:LogoutRequest> and <samlp:LogoutResponse> elements received by this SP must be signed.</p>
<p>[CONFIG] onelogin.saml2.security. want_assertions_signed = false</p>	<p>Whether the <saml:Assertion> elements received by this SP must be signed.</p>
<p>[CONFIG] onelogin.saml2.security. sign_metadata =</p>	<p>Whether the Metadata of this SP must be signed. Use either <code>null</code> (for not signing) or <code>true</code> (sign using SP private key).</p>
<p>[CONFIG] onelogin.saml2.security. want_assertions_encrypted = false</p>	<p>Whether the Assertions received by this SP must be encrypted.</p>
<p>[CONFIG] onelogin.saml2.security. want_nameid_encrypted = false</p>	<p>Whether the <code>NameID</code> received by this SP must be encrypted.</p>

<p>onelogin.saml2.security. requested_authncontext = urn:oasis: names:tc:SAML:2.0:ac:classes:Password</p>	<p>Authentication context. If empty, no AuthContext is sent in the AuthNRequest. Accepts one or more comma-separated values.</p>
<p>onelogin.saml2.security.onelogin.saml2. security. requested_authncontextcomparison = exact</p>	<p>Allows the <code>authn</code> comparison parameter to be set. Defaults to 'exact'.</p>
<p>onelogin.saml2.security. want_xml_validation = true</p>	<p>Whether the SP validates all received XML.</p> <p><i>Required:</i> <code>onelogin.saml2.strict = true</code> must be set to validate the XML.</p>
<p>onelogin.saml2.security. signature_algorithm = http://www.w3.org/2000/09/xmldsig#rsa-sha1</p>	<p>Algorithm that the toolkit uses for the signing process. Options:</p> <ul style="list-style-type: none"> • http://www.w3.org/2000/09/xmldsig#rsa-sha1 • http://www.w3.org/2000/09/xmldsig#dsa-sha1 • http://www.w3.org/2001/04/xmldsig-more#rsa-sha256 • http://www.w3.org/2001/04/xmldsig-more#rsa-sha384 • http://www.w3.org/2001/04/xmldsig-more#rsa-sha512
<p>Organization and Contacts</p>	
<p>onelogin.saml2.organization.name = onelogin.saml2.organization. displayname = onelogin.saml2.organization.url = onelogin.saml2.organization.lang =</p> <p>onelogin.saml2.contacts.technical. given_name = onelogin.saml2.contacts.technical. email_address = onelogin.saml2.contacts.support. given_name = onelogin.saml2.contacts.support. email_address =</p>	

Gateway Access Control Policies

The Content Gateway provides a rich access control mechanism that allows for coarse to fine-grained control over user access to content within a storage domain and administrative actions within the management API. Access control is defined within Policy documents that may specify permissions on specific objects when necessary. Access control Policy documents are stored in the following locations:

- Root Policy file
- Tenant Policy sub-resource
- Storage domain Policy sub-resource
- Bucket Policy sub-resource

The root Policy document is stored in a JSON file:

```
/etc/caringo/cloudgateway/policy.json
```

This file *must* be kept synchronized between all Gateway servers. Changes to the local file take effect without the need to restart the Gateway. The policy sub-resource for tenants, storage domains, and buckets is kept within the cluster, is shared among all Gateway servers, and is accessed through the management API or storage API.

Note
 The root Policy configuration file must exist and must contain a valid JSON string or be blank. The minimum valid JSON content is "{}".

- [Policy Evaluation](#)
- [Modifying Policies](#)
- [Policy Document](#)

Policy Evaluation

- [Administrator Roles](#)
- [Object Ownership](#)
- [Evaluation Precedence](#)

These are the administrator roles and the rules for the order of policy evaluation. Administrator roles are based on ownership and access permissions defined within the Gateway and storage contexts. While these are all users, possibly in different LDAP DNs and/or PAM, and can all be the same set of users, the following definitions are useful for describing the normal responsibilities that each classification of administrator has. They are normal boundaries where access control is segregated.

(i) Role-based Access Control

While Swarm only defines the role of owner, role-based access control (RBAC) definitions with varying complexity can be created as required for the organization using the Gateway's access control policies. The Cluster, Tenant, Domain, and Bucket "administrators" are common roles that can be used, but they are not required or hard-coded into the system.

Administrator Roles

Type and Administrator Role	Example
<p>CLUSTER</p> <p><i>Define who can create tenants and non-tenanted domains</i></p> <p>Analogous to the Swarm administrator except they are defined in an external identity management system. This user or group is specified in the policy.json root Policy configuration file.</p>	<p>This policy.json file defines and grants full permissions to the cluster administrators group called ClusterAdmins. The members of the ClusterAdmins group are the cluster administrators users and are often the same that maintaining the physical infrastructure.</p> <pre data-bbox="520 1136 954 1631"> { "Version": "2008-10-17", "Id": "ClusterAdminsPolicy", "Statement": [{ "Sid": "1", "Effect": "Allow", "Principal": { "group": ["ClusterAdmins"] }, "Action": ["*"], "Resource": "/*" }] }</pre>

<p>TENANT</p> <p><i>Define who can create domains for the tenant</i></p> <p>Owner of the tenant object as specified by the X-Owner-Meta metadata header. It is common for the tenant administrator to create a Policy document for the tenant that grants permissions for a group of users to act on the same authority of the tenant administrator.</p>	<p>This tenant Policy document grants full access to a group called TenantAdmins whose members come from users within the acme tenant.</p> <pre>PUT /_admin/manage/tenants/acme/etc/policy.json { "Version": "2008-10-17", "Id": "TenantAdminsPolicy", "Statement": [{ "Effect": "Allow", "Sid": "1", "Principal": { "group": ["TenantAdmins"] }, "Action": [""], "Resource": "/" }] }</pre>
<p>DOMAIN</p> <p><i>Define who can create buckets and unnamed objects</i></p> <p>Owner of the storage domain as specified by the X-OwnerMeta metadata header. It is common for the domain administrator, owner of the storage domain, to create a Policy document for the domain that grants permissions for a group of users to act on the same authority of the domain administrator.</p>	<p>This domain Policy document grants full access to a group called DomainAdmins whose members come from users within the domain.</p> <pre>PUT http://DOMAIN/?policy { "Version": "2008-10-17", "Id": "DomainAdminsPolicy", "Statement": [{ "Effect": "Allow", "Sid": "1", "Principal": { "group": ["DomainAdmins"] }, "Action": [""], "Resource": "/" }] }</pre>

BUCKET

Define who can create named objects within the bucket

Owner of the bucket as specified by the **X-Owner-Meta** header. The bucket administrator, owner of the bucket, can attach a Policy document to the bucket that defines the access control policy for the bucket and its contents.

This bucket policy grants any authenticated user full access under `http://DOMAIN/mybucket/incoming/*` and grants users in the **Finance** group full access under `http://DOMAIN/mybucket/reports/*`.

```
PUT http://DOMAIN/mybucket?policy
{
  "Version": "2008-10-17",
  "Id": "MyBucketPolicy",
  "Statement": [
    {
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
        "user": [
          "*"
        ]
      },
      "Action": [
        "*"
      ],
      "Resource": "/mybucket/incoming"
    },
    {
      "Sid": "2",
      "Effect": "Allow",
      "Principal": {
        "group": [
          "Finance"
        ]
      },
      "Action": [
        "*"
      ],
      "Resource": "/mybucket/reports"
    }
  ]
}
```

All objects are contained with the bucket context **mybucket**. The access control policy matches named objects with the prefixes **incoming/** and **reports/** within that bucket.



Note

Notice the bucket name is included when specifying resources in the bucket policy.

Object Ownership

All objects created and updated through the Gateway have the **X-Last-ModifiedBy-Meta** metadata value set to the authenticated user performing the request. The value is blank if an object is created by an anonymous user. Additionally, unless the request includes an **X-Owner-Meta** metadata value, it is assigned the value of the authenticated user or blank for anonymous. If **X-Owner-Meta** is blank, everyone is considered to be the owner for policy evaluation.

- For users authenticated within the *same* domain as the objects, the format of the metadata values is the base user name. For example: "john".
- For users authenticated from a *different* domain than the objects, the format is fully qualified with the authenticating domain. For example: "john@[another.com](#)".

- For users authenticated from the *root* IDSYS, the format is user + "@". For example: "john@".
- If an object is created by a cluster administrator using an administrative override, the format is the same since the user is authenticated from the root IDSYS. For example "admin@".

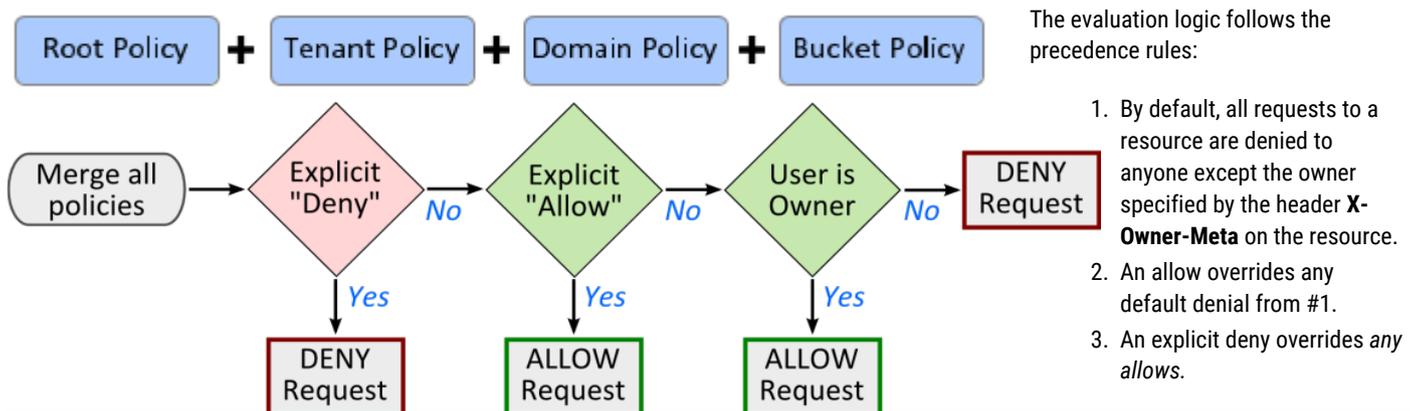
See [Modifying Policies](#).

Evaluation Precedence

When evaluating a user's authorization to perform a requested action, all relevant Policy documents are first merged together. Policies can exist at the root, tenant, storage domain, and bucket levels and, depending upon the request's level within the API hierarchy, one or more of these policies are merged together for evaluation.

For example, if a user requests to read an unnamed object within a storage domain, the Root Policy, Tenant Policy, and Domain Policy are merged together. Requesting the read of a named object within a bucket also merges the Bucket Policy along with the root, tenant, and storage domain policies.

This is the policy evaluation logic:



Note

The order in which the policies are evaluated has no effect.

The result of an explicit deny overriding allow is that if an action is denied to all users (`"user" : ["*"]`), even if also explicitly allowing oneself the same action, one is denied the ability to perform the action.

If a request is for a non-existent domain or bucket context object, policy evaluation is short-circuited and an HTTP 404 response code is returned to the client. When the domain or bucket exists, policy evaluation returns an HTTP 403 if the user is not authorized to perform the action. The practical implication is a user authorized to access a storage domain may be able to detect the existence of a bucket for which they do not have access. This paragraph only applies to context objects and not to objects that hold user content.

If the **X-Owner-Meta** header is blank or missing from a storage domain or bucket object, its ownership is anonymous and all users match as the owner. The result is that everyone has owner permissions on the storage domain or bucket. It is a best practice to always assign ownership to context objects.

Modifying Policies

- [Important](#)
- [Administrative Override](#)
- [Domain IDSYS and Bucket Policy Overrides](#)
- [User Name Login Formats](#)
- [Cross-Tenant Access Control](#)
- [Swarm Node Status Page Visibility](#)

The root Policy configuration is stored in the policy.json file on the Gateway server's disk so it is always available and so an administrator can always modify it. Combined with an administrative login the cluster administrator cannot be locked out of the storage cluster.



Important

It is crucial the root Policy document is synchronized across all servers when more than one Gateway server is deployed.

See [Defined ETC Documents](#) for modifying a Policy for a tenant, storage domain, or bucket through the management API.

See [SCSP Context Sub-resources](#) for modifying the policy sub-resource for a domain or bucket through storage API.

Administrative Override

The Gateway's access control mechanism has a provision to allow cluster administrators to log in to a tenant or storage domain and bypass all restrictions contained within that tenant's or domain's policy sub-resource. This can be used to access content where the owner mistakenly cut-off all access and locked out users. Another example is if the IDSYS for the storage domain has incorrect information within preventing all user logins.

The cluster administrator performs an administrative bypass request by putting an exclamation point "!" before and an at sign "@" after the username. The form of the user name portion of the login is: "!" + name + "@"

This instructs Gateway to use only the root IDSYS and root Policy documents when looking up the user and when evaluating whether an action is allowed. Any idsys or policy sub-resource associated with the tenant, storage domain, or buckets are ignored.

This example logs in as the user named admin and blanks the Policy document on the `marketing.example.com` domain so only the owner identified by the X-OwnerMeta header of the domain object is allowed to operate in the domain.

```
curl -u '!admin:adminpw' -X PUT
-H 'Content-type: application/json'
-d '' 'http://DOMAIN/?policy'
```

Presumably, after unlocking the storage domain, the owner re-submits a corrected Policy document allowing the proper user access for the domain's content.

Domain IDSYS and Bucket Policy Overrides

Along the lines of the administrative override, Gateway provides a mechanism for accessing a storage domain by bypassing either the domain's IDSYS or a bucket's Policy.

The user name for the request uses the form: user + "@". For example "psmith@" to bypass the storage domain's IDSYS in favor of the root IDSYS. Requests performed using user names in the form "user@" are still subject to the domain and bucket Policy. Logins in this form only affect the authentication source for users.

The user name for the request uses the form: "!" + user, such as "!psmith" or "!psmith@other.com" to bypass a bucket's Policy. Requests of this form are authenticated using a domain IDSYS and are still subject to the domain Policy. This form of login can be used by the domain administrator to modify a the bucket Policy for another user's bucket. Notice this override also works when the domain owner is from another tenant domain.

User Name Login Formats

These are the meanings for the different user name formats used to authenticate with Gateway.

User Name	Effect	Restrictions
user	Use this domain's IDSYS and domain + bucket Policy sub-resource.	User must be able to log in to this domain's IDSYS.
user@otherdomain	Use other domain's IDSYS and this domain + bucket Policy sub-resource.	Other domain must exist in same storage cluster as this domain. User must be able to log in to other domain's IDSYS.
user+othertenant	Use other tenant's IDSYS and this domain + bucket Policy sub-resource	Other tenant must exist in same storage cluster as this domain. User must be able to log in to other tenant's IDSYS.
!user	Use this domain's IDSYS and domain Policy; ignore any bucket Policy.	Only domain owner can use this.
!user@otherdomain	Use other domain's IDSYS and this domain's Policy; ignore any bucket Policy.	Only domain owner can use this; owner is from another domain.
!user+othertenant	Use other tenant's IDSYS and this domain's Policy; ignore any bucket Policy.	Only domain owner can use this; owner is from another tenant.
user@	Use root IDSYS (ignore domain IDSYS) and use domain + bucket Policy sub-resource.	User must be able to log in to root IDSYS.
!user@	Use root IDSYS (ignore domain IDSYS) and root Policy (ignore domain + bucket Policy sub-resource).	User must be able to log in to root IDSYS.

Cross-Tenant Access Control

The access control policy evaluation in Gateway allows for the specification of users and groups that exist in other tenants and other storage domains within the Swarm cluster. This is performed by appending @domain or +tenant qualifications onto a user or group name.

See "Qualification of User/Group Names" in [IDSYS Document Format](#) for a full explanation for user and group qualification.

While there is no limit on the number of other domains specified in a Policy document, all tenants and storage domains must exist within the same storage cluster.

Applications should not use `@domain` and `+tenant` qualifications unless the users and groups to which the principals refer are actually outside of the storage domain or tenant in which the policy resides. An `IDSYS` must exist within the referenced tenant or storage domain when users and groups are fully qualified.

Swarm Node Status Page Visibility

The Swarm node status page of the legacy Admin Console is presented in response to a "GET /" request. If access to the resource `/*` is granted to anonymous or any user, the result is they are authorized to request `/` and are able to view the node status page for the back-end storage cluster. Note: due to connection pooling done by the Gateway, there is no way to specify which storage node's status page is returned on a request.

The following addition to the policy for the domain explicitly denies anonymous users access to `/` so they cannot view the node status page.

```
...
{
  "Effect": "Deny",
  "Sid": "AdminPage",
  "Principal": {
    "anonymous": [
      "*"
    ]
  },
  "Action": [
    "GetObject"
  ],
  "Resource": "/"
}, ...
```

By changing the principal, the previous policy example can be adapted to deny access to the node status page for any user or group.

Policy Document

- [Policy Document Fields](#)
- [Policy Format](#)
- [Principals](#)
- [Request Actions](#)
- [Policy Conditions](#)



Best practice

Each **Action** domain administrators are allowed to perform can be listed to create a policy for them having broad permissions. It is easier and less error-prone to list the few actions they *cannot* perform (such as **DeleteDomain** and **CopyDomain**) by replacing the entire **Action** statement with the **NotAction** statement:

```
{ "Version": "2016-10-17", "Statement": [ { "Sid": "Grant all except excluded domain
operations to admins2", "Resource": "/*", "Effect": "Allow", "Principal": { "group": [
"admins2" ] }, "NotAction": [ "CopyDomain", "DeleteDomain" ] } ] }
```

Policy Document Fields

Policy documents are JSON-formatted objects.

Field	Sub-field	Description
Version		optional; if used must be "2012-10-17" or "2008-10-17" to match S3
Id		optional name to describe the policy
Statement		a list of rules
	Sid	unique identifier for each statement
	Effect	whether the rule allows or denies an operation
	Principals	defines users and groups to which the rule applies
	Action	content operation (request)
	NotAction	logical inverse of Action; supports all Action values
	Resource	path for which the rule applies; "/" prefix is optional
	Condition	conditional requirements for the request

Policy example

```
{
  "Version": "2008-10-17",
  "Id": "Comics-and-Superheros",
  "Statement": [
    {
      "Effect": "Allow",
      "Sid": "1",
      "Principal": {
        "group": [
          "Development",
          "QA",
          "testers@partner.net"
        ],
        "user": [
          "jdoe",
          "gcarlin@funny.com",
          "tony+starkindustries"
        ]
      },
      "Action": [
        "GetObject"
      ],
      "Resource": "/*",
      "Condition": {
        "StringLike": {
          "Referer": [
            "*example.com"
          ]
        }
      }
    }
  ]
}
```

Policy Format

The format of Gateway's Policy document is modeled after Amazon S3 bucket policies. There is one policy document per tenant, per storage domain, and per bucket. Updating a policy document completely replaces any existing document. The root Policy plus the relevant tenant, storage domain, and bucket policies are all first merged together when evaluating a request.

The policy is a set of rules specifying the conditions under which a request is allowed or denied within the storage API or the management API (such as create a domain, bucket, or object). Each statement in a policy specifies the principals allowed or denied the ability to perform an action on a resource. These are examples of each of these elements of a policy specification.

- **Principals:** user "jdoe", group "Development"
- **Action:** GetObject, DeleteObject
- **NotAction:** CopyDomain, DeleteDomain
- **Resource:** "mybucket/private/*", "mybucket/photos/picture01.jpg"
- **Condition:** "StringLike" : { "Referer" : ["*example.com"]} }
- **Effect:** "Allow", "Deny"



Best practice

Make the Policy documents as small as reasonable for best performance. Performance is affected because the statements need to be evaluated for each request to check if the resource matches. Choose the most efficient definitions; **NotAction** lists may be shorter than **Action** lists.

The `Id` field is provided for end-user use only and is ignored by the policy evaluation. The `Version` field is optional and the sole valid value is "2008-10-17" if defined.



Caution

Do not prefix resources with "arn:aws:s3:::" and actions with "s3:" strings: these are ignored by the policy evaluation and negatively impact performance.

Principals

Principals are users or groups to whom the access control statement applies. This includes the specification of an anonymous user that has not authenticated. The format is:

Specifying authenticated users

```
"{user|group}": [{list- of- principals}]
```

These notation formats are recognized for anonymous principals:

Anonymous principals

```
Principal: "*"
Principal: { "anonymous" : [ "*" ] }
Principal: { "anonymous" : "*" }
Principal: { "AWS" : [ "*" ] }
Principal: { "AWS" : "*" }
```

This is an example that includes users and groups from other storage domains and tenants.

Including users and groups

```
"Principal" : {
  "user": [ "vcerf", "timbl", "ltorvalds@kernel.org" ],
  "group": [ "Development", "QA+acme", "Development+acme" ],
}
```

See *Cross-tenant Access Control* below for the meaning of the @domain and +tenant suffixes for users and groups.

The following table explains the meaning of the different forms of the principal specifications.

Principal	Description
"anonymous":["*"]	An anonymous, unauthenticated user
"user":["*"]	Any authenticated user from any IDSYS scope
"user":["*@austin"]	
"user":["*+texas"]	
"user":["gcarlin"]	A user named 'gcarlin' from this scope's IDSYS (or inherited IDSYS if applicable). This is a non-qualified user name since no domain, tenant, or root scope is specified.
"user":["gcarlin@cars"]	A user named 'gcarlin' from the 'cars' storage domain's IDSYS (or its inherited IDSYS if applicable)
"user":["gcarlin+movies"]	A user named 'gcarlin' from the 'movies' tenant's IDSYS (or its inherited IDSYS if applicable)
"user":["gcarlin@"]	A user named 'gcarlin' only from the root IDSYS
"group":["admins"]	Any member of the group named 'admins' from this scope's IDSYS (or inherited IDSYS if applicable). This is a non-qualified group name since no domain, tenant, or root scope is specified.

"group": ["admins@hockey"]	Any member of the group named 'admins' from the 'baseball' storage domain's IDSYS (or its inherited IDSYS if applicable)
"group": ["admins+sports"]	Any member of the group named 'admins' from the 'sports' tenant's IDSYS (or its inherited IDSYS if applicable)
"group":["admins@"]	Any member of the group named 'admins' only from the root IDSYS

In many cases it is unnecessary to explicitly grant permissions to the user named by the `X-Owner-Meta` header, of a tenant, storage domain, or bucket because they are granted permission for all operations by default. This default permission means that careful consideration should be given when assigning ownership to another user – especially for tenants or storage domains.



Linux root User

The `root` user (uid=0) is not allowed to authenticate to Gateway using Linux PAM.

User and group principals can be written as fully-qualified or non-qualified within Policy documents. A fully-qualified principal is one that defines the domain, tenant, or root scope (example: "gcarlin@cars"). A non-qualified principal does not include the scope for the principal (example: "gcarlin"). The scope of a principal defined the starting point for looking for the first available IDSYS definition with which to look for the principal. The starting point for the IDSYS search starts from the point of the resource being requested if the principal is non-qualified.

The IDSYS precedence model allows the root scope to be overridden by the tenant and the domain scopes. The search order when looking for the first available IDSYS is to search backwards from domain to tenant to root. The Gateway looks for the first IDSYS that is defined at the domain, then the tenant, then at the root if a group is specified as "admins@hockey" for requests within the "hockey" storage domain. The group "admins" from only that IDSYS are used upon finding the first IDSYS definition.

Request Actions

Policy documents are used to control actions with both the Management API and the Storage API. The policy actions that apply to a given request depends upon the API being used, management or storage. The list of Policy documents that are merged together for policy evaluation is determined by the path in the API hierarchy being accessed.

Request actions are included in the Effect field result when listed within an Action field. The listed actions are explicitly denied if the effect is "Deny". Request actions are excluded from the Effect field results when listed within a `NotAction` field. All actions other than those listed are explicitly denied if the effect is "Deny".

Policy Actions for Administration (Management API)

These are the Management API actions that can be controlled within a Policy document and the applicable scope where the actions are used. If an action is granted or denied within a scope where not applicable, it has no effect.

Scope indicates where the policy action is applicable: (R) root, (T) tenant, (D) domain, (B) bucket.

Manage	Action	Scope	Description
Global	*	R,T,D,B	all actions
Tenants	ListTenants	R	List all tenants
	CreateTenant	R	Create a new or change an existing tenant
	GetTenant	R,T	Retrieve tenant properties
	DeleteTenant	R,T	Permanently remove tenant properties
	ListEtc	R,T	List documents associated with a tenant
Domains	ListDomains	R,T	List the domains owned by the <code>_system</code> tenant
	CreateDomain	R,T	Create a domain for the <code>_system</code> tenant
	GetDomain	R,T,D	GET a domain
	DeleteDomain	R,T,D	Delete a domain
Policies	ListEtc	R,T,D	List documents associated with a tenant or a storage domain
	PutPolicy	R,T,D	Create or update an access control policy JSON document
	GetPolicy	R,T,D	Read an access control policy JSON document
	DeletePolicy	R,T,D	Permanently remove an access control policy JSON document
Authentication Tokens	TokenAdmin	R,T,D	Create and list authorization tokens for other users in the same scope
	CreateToken	R,T,D	Create an authentication token
	ListTokens	R,T,D	List user authentication tokens

	ValidateToken	R,T,D	Read an authentication token
	DeleteToken	R,T,D	Delete an authentication token

Policy Actions for Content (Storage API)

This table defines the Storage SCSP API actions that can be controlled within a Policy document and the applicable scope where the actions are used. An action has no effect if granted or denied within a scope where not applicable.

See [Specifying Permissions in a Policy](#) in the AWS documentation for Amazon S3 operation permissions.

Scope indicates where the policy action is applicable: (R) root, (T) tenant, (D) domain, (B) bucket.

Storage	Action	Scope	Description
Global	*	R,T,D,B	all actions
Quotas	PutQuota	R,T,D,B	Create or update quota configuration settings on a tenant, domain, or bucket. This is not granted by "*"; it must be granted explicitly.
	GetQuota	R,T,D,B	Retrieve quota configuration settings on a tenant, domain, or bucket. This is not granted by "*"; it must be granted explicitly.
Objects	GetObject	R,T,D,B	Retrieve an object and its metadata
	GetObjectAcl	R,T,D,B	Retrieve the Access Control List (ACL) settings on an object
	CreateObject	R,T,D,B	Add a new object and metadata to a domain (unnamed) or bucket (named). There is no S3 equivalent, and it is permissioned separately here because Swarm distinguishes POST and PUT.
	AppendObject	R,T,D,B	Append to an object
	CopyObject	R,T,D,B	Update metadata on an object while keeping the same object name
	PutObject	R,T,D,B	Update an object PutObject, unlike S3, is for overwriting an existing object, not creating new one (CreateObject).
	PutObjectAcl	R,T,D,B	Set an object's access control list
	DeleteObject	R,T,D,B	Permanently remove an object and its metadata
Buckets	CreateBucket	R,T,D	Create a bucket
	PutBucket	R,T,D	Synonym for CreateBucket
	PutBucketAcl	R,T,D	Set bucket access control list
	PutBucketCORS	R,T,D,B	Set CORS configuration on a bucket
	GetBucket	R,T,D,B	GET or HEAD a bucket
	GetBucketAcl	R,T,D,B	Get bucket access control list

	GetBucketCORS	R,T,D,B	Get CORS configuration on a bucket
	CopyBucket	R,T,D,B	Update the metadata on a bucket while keeping the same bucket name
	DeleteBucket	R,T,D,B	Delete a bucket
	ListBucket	R,T,D,B	List and search the objects contained in a bucket
Bucket Sub-Resources	PutPolicy	R,T,D,B	Create or update a policy, xform, sub-resource for a bucket. While an owner of bucket can always overwrite the policy, this allows a group to be granted this permission.
	GetPolicy	R,T,D,B	GET or HEAD the policy, xform, sub-resource for a bucket
	DeletePolicy	R,T,D,B	Permanently remove a policy, xform, sub-resource for a bucket
Domains	CreateDomain	R *	Create a domain
	GetDomain	R,T,D	GET or HEAD a domain
	CopyDomain	R,T,D	Update the metadata on a domain while keeping the same domain name
	DeleteDomain	R,T,D	Permanently remove a domain
	ListDomain	R,T,D	List and search all objects (buckets and unnamed objects) within the domain
	ListDomains	R *	List all domains that exist in the cluster
Domain Sub-Resources	PutPolicy	R,T,D	Create or update a policy, xform, idsys, sub-resource for a domain
	GetPolicy	R,T,D	GET or HEAD policy, xform, idsys, sub-resource for a domain
	DeletePolicy	R,T,D	Permanently remove a policy, xform, idsys, sub-resource from a domain
Authentication Tokens	CreateToken	R,T,D	Add a new authentication token
	ListTokens	R,T,D	Search for and list authentication tokens
	ValidateToken	R,T,D	Read and verify an authentication token
	DeleteToken	R,T,D	Permanently remove an authentication token

i * The Tenant scope does not apply to SCSP CreateDomain and ListDomains because these operations refer to the **_system** tenant implicitly: there is no "tenant" access through SCSP.

Policy Conditions

The Condition field of the Policy document allows for conditions other than user and action to be placed upon the request.

Conditionals in a Policy document allow for the value matching functions:

`StringEqualsIgnoreCase`
`StringLike`

- `StringEqualsIgnoreCase` matching is an exact match ignoring letter case.
- `StringLike` matching allows for glob-style pattern matching for the values and ignores letter case.

All matching conditionals are evaluated on each request similar to the way paths are matched to resources.

Referral Condition

Gateway supports conditional requirements on the value of the HTTP/1.1 `Referer` header.



Note

Gateway uses the same spelling (or misspelling) of "referer" to match the header specification in [RFC 7231 5.5.2](#).

The `Referer` header is commonly provided by web browsers when retrieving content referenced by an HTML document. While the browser, or HTTP client library, is free to provide any value they wish for this header, it is commonly used to detect the source reference for a resource request. The following logical evaluation takes place for the referral matching condition:

- Policy statements without `Referer` conditions are considered for authorization if there is no `Referer` header in the request.
- Policy statements without `Referer` conditions are considered for authorization if there is a `Referer` header in the request.
- The policy statement is considered for authorization if the values match if there is a `Referer` header in the request and there are policy statements with `Referer` conditions. Values are matched as follows:
 - The condition matches if any value matches for the condition-key if there are multiple values in a conditional value list. It is a logical OR match.
 - All conditions must match if there are multiple condition-types in a condition-block. It is a logical AND match.

Rules that match a request are chosen from available rules according to Resource, Action, referral Conditions as described. Among those rules, further filtering is performed by Principal and lastly according to deny/apply semantics. Note: referral conditions are processed in all actions.

This example shows a conditional restriction that the `Referer` value must be `example.com` or `another.com`.

```
{
  "Statement" : [
    {
      "Condition" : {
        "StringEqualsIgnoreCase" : {
          "referer" : ["example.com", "another.com"],
          ...
        }
      }
    }
  ]
}
```

This example shows a conditional restriction that uses the `StringLike` match to restrict the `Referer` value to any subdomain within a parent domain.

```
{
  "Statement" : [
    {
      "Condition" : {
        "StringLike" : {
          "referer" : [ "*example.com" ],
          ...
        }
      }
    }
  ]
}
```

While `Referer` header restrictions are commonly used to prevent bandwidth stealing due to cross-linking by unaffiliated web sites, the requesting client is being trusted to accurately populate this header. `Referer` header restrictions work well to prevent unauthorized cross-linking, this is not a mechanism to be relied upon to provide site security.



Note

The referral conditions apply to the Storage API and not to the Management API operations.

Prefix Condition

Gateway supports conditional requirements on the prefix match. This can be used to require object searching be constrained to a set of prefix patterns.

This is an example of a policy that only allows the listing of objects if a prefix query argument is specified.

```
{
  "Sid": "AllowListingOfSharedFolder (GET /mybucket?prefix=home/Shared/)",
  "Action": [
    "ListBucket"
  ],
  "Effect": "Allow",
  "Principal": { "group": "Editors" },
  "Resource": [
    "mybucket"
  ],
  "Condition": {
    "StringLike": {
      "prefix": [
        "home/Shared/"
      ]
    }
  }
}
```

Allowed: `GET /usersbucket?format=json&prefix=home/Shared/`

Denied: `GET /usersbucket?format=json&prefix=home/Shared/`

Gateway Troubleshooting

This section provides a starting point for troubleshooting operational issues with your Gateway deployment and integrated applications.

- [Troubleshooting Gateway and Elasticsearch](#)
- [Bad IDSYS or Policy](#)
- [Unexpected HTTP Responses](#)
- [LDAP Configuration](#)
- [Domain and Bucket Ownership](#)
- [Checking the Gateway Service](#)
- [Portal Upload Error](#)
- [Unexpected HTTPS to HTTP Redirects](#)

Troubleshooting Gateway and Elasticsearch

Note how Gateway 5.4 and higher operates when Elasticsearch is down when troubleshooting Gateway with Elasticsearch:

- Gateway starts even if the Elasticsearch cluster is down or in a red state.
- Operations requiring Elasticsearch fail, such as listings and graphs for metering and metrics.
- Reads and writes succeed.
- The Storage UI continues to work.
- Gateway periodically checks for Elasticsearch, and full functionality resumes once Elasticsearch is back up and fully initialized.

Log in the Storage UI, and view the Elasticsearch section on the Dashboard for real-time status information about the Elasticsearch cluster.

See [how to fix common Elasticsearch cluster issues](#) and **Reports > Elasticsearch** page for detailed troubleshooting.

See **Elasticsearch Reports** in [Using Cluster Reports](#).

Bad IDSYS or Policy

If you write an incorrect IDSYS to a tenant or a storage domain, subsequent attempts to access the system returns an HTTP 503 error. This is an example response:

```
HTTP/1.1 503 Service Unavailable
Server: CASTor Cluster/6.5.4
Via: 1.1 172.16.99.70 (Cloud Gateway SCSP/2.2)
Gateway-Request-Id: D9DF0347CB7EAAE9
Gateway-Error-Message: Unable to connect to identity system
    ldap://172.16.99.20:636 as cn=gateways,dc=caringo,dc=com:
    javax.naming.ServiceUnavailableException: 172.30.0.42:636; socket closed
Content-Length: 44
Identity system failure or misconfiguration
```

To work around this, authenticate as qualified user that is defined in the tenant IDSYS or in the root IDSYS and replace the bad IDSYS. For example, if the user `admin` exists in the root IDSYS, write a corrected version of the storage domain's IDSYS and authenticate the request as user `"admin@"`.

 **Warning**

If you write an incorrect Policy to a tenant or storage domain, you can lock yourself out.

Workaround

To work around a Policy problem, authenticate with a "!" prefix on the user name and replace the bad Policy. For example, if the user `admin` exists in the root IDSYS and the storage domain's Policy has denied access to all users, write a corrected version of the Policy to the storage domain by authenticating as `"!admin@"`.

For more information, see the [IDSYS Document Format](#).

Unexpected HTTP Responses

If you get unexpected HTTP response codes while using or integrating with Gateway, use these tips to troubleshoot the cause of the responses. An example of an unexpected response is if permission is denied (HTTP 403) on an object believed to be accessible.



Tip

Enabling [Swarm Storage audit logs](#) allows tracking all requests the Gateway sends related to a client request.

- [Tip](#)
- [Request ID in responses](#)
 - [SCSP response header](#)
 - [S3 response header](#)
 - [S3 error response body](#)
- [Request ID in logs](#)
 - [Log level](#)
 - [Tip](#)

Request ID in responses

To aid with tracking transactions, all Gateway HTTP responses contain a header with the request ID. This request ID is also recorded in the Gateway's server log file and the audit log file. Searching the server log file for a given request ID shows the processing steps that took place during the handling of the request.

These are three examples where the request ID is found in a client response:

SCSP response header

```
Gateway-Request-Id: 375400C95338546F
```

S3 response header

```
x-amz-request-id: 375400C95338546F
```

S3 error response body

```
<RequestId>375400C95338546F</RequestId>
```

Request ID in logs



Log level

You must enable debug-level logging on Gateway to see these log entries. See [Gateway Logging](#).

Search for the request ID in the Gateway server log:

```
grep "375400C95338546F" /var/log/cloudgateway/server.log
```

The search results from the log show:

- Request URI and whether an Authorization or cookie header is on the request
- Action being performed, such as **CreateDomain**, **GetObject**
- Owner of the context for the request
- Merged Policy document used to evaluate authorization
- LDAP search filter used for user or group lookups
- Reason for the HTTP response

The merged Policy is normally a combination of the root, domain, and bucket policies. An example log entry showing the context owner and a merged Policy document is:

```
2014-03-31 11:12:32,442 DEBUG [qtp1994043452-35|C66CF2A1D4DD4C8D]
Auth: AUTHENTICATING: 'ldap john@'
Action is GetObject, user idsys is ldap (root), context owner is john@ and merged policy is:
[Sid=1 Allow [AllActions] "/" "*" {group=[CloudAdmins]} {}]
```

When using the Policy conditions, such as the **Referer** header restrictions, the merged Policy that is logged is the one used to evaluate permissions for the request. Check the condition statements to see which is being used if the expected portion of the Policy is not displaying.

Additional error details are contained with the HTTP response header `Gateway-ErrorDetails` and are also logged in the Gateway server log. An example of this type of log message is:

```
2012-10-19 08:41:28,327 DEBUG [qtp596850781-35 - /reports?domain=example.com| F09B3F5FCA0A477F]
Auth: Request failed: 403 User is not allowed and is not owner.
owner: john, user: george, dn: uid=george,ou=people,dc=example,dc=com
```

In the previous example, the user **george**, the full LDAP DN is given, is not allowed to perform the requested action within the **/reports** bucket because he is not the owner, **john**, and because there is no Policy that grants him permission.

If necessary for application debugging, the Gateway can dump the HTTP request headers received with each request. To enable request header logging, add the following setting to the `gateway.cfg` file and restart the Gateway process:

```
[debug]
showRequest = true
```

 **Tip**

Use debug sparingly! It can produce a significant amount of extra information in the server log, including security-related **Authorization** and **Cookie** headers.

An example of the request headers in the log output is:

```
2013-08-30 15:23:24,804 DEBUG [qtp1872474714-40|F69277697D792B98]
Auth: REQUEST: POST /?domain=john.example.com AUTHORIZATION
ContentType: application/castorcontext
workaround-content-type: application/ castorcontext
Host: 172.10.8.5:8084
Content-Length: 0
Accept: */ *
User-Agent: curl/7.21.4 (universal-apple-darwin11.0) libcurl/7.21.4 OpenSSL/0.9.8x zlib/1.2.5
Authorization: Basic am9objpwYXNzd29yZA==
```

LDAP Configuration

Problems with the configuration of the LDAP identity management settings can prevent user authentication and the determination of group membership. All LDAP configuration items are kept within the root IDSYS document, stored in the Gateway server's file system, and the IDSYS documents for tenants and storage domain. Start troubleshooting by:

1. Determining which IDSYS document is being used based on the format of the user name (see [Content Application Development](#) for details on the login format).
2. Verifying the fields in the IDSYS being used are correct.

After determining the IDSYS that is being used, debug basic connectivity and queries directly with the LDAP server. Look in the Gateway server's log to get the exact LDAP search filter it is trying to use.

This is an example from the log where it checks if a user belongs to an allowed group:

```
2012-09-13 22:25:47,671 DEBUG [qtp1355087478-37 - /1347593126.86weirdbucket/x/foo.txt?domain=1347
Policy: Searching for user in ou=groups,dc=example,dc=com
with filter (&(objectclass=*) (memberUid=john)(|(cn=Finance)))
```

The log entry of the LDAP search filter can be converted to an LDAP URL that can be used by a tool like cURL to query the LDAP server. The format of the URL is:

```
ldap://HOST:PORT/ROOT??sub?FILTER
```

Using the LDAP search filter information from the example Gateway log, this shows how to use the cURL command to query the LDAP server directly.

```
curl "ldap://localhost/ou=groups,dc=example,dc=com??sub?(&(objectclass=*) (memberUid=john)(|(cn=F
```

If the connection is successful and the query finds users with the group, the output is similar to this:

```
DN: cn=Finance,ou=groups,dc=example,dc=com
gidNumber: 10002
memberUid: fred
memberUid: john
description: Group account
objectClass: posixGroup
cn: Finance
```

If there are errors, resolve them and update the IDSYS document with the corrections.

Domain and Bucket Ownership

- [Unknown Tenant](#)
- [Note](#)
 - [Retrieve domain metadata](#)
 - [Change the domain owner](#)
 - [Retrieve a bucket object's metadata](#)
 - [Change the bucket](#)
 - [Change to anonymous ownership](#)

It can be helpful to query Swarm directly to troubleshoot access permission problems related to ownership for domain or bucket contexts. Check the context (domain or bucket) owner has the correct value.

i Unknown Tenant

Gateway handles the request as if it was from the "System Tenant" and provides troubleshooting guidance in the application logs if Gateway receives a request to a domain marked as belonging to an unknown tenant.

i Note

The string `{node-IP}` represents the IP address of any Swarm node in the storage cluster. These examples assume direct access to the storage nodes is possible without going through the Gateway.

Retrieve domain metadata

```
curl -I -L 'http://{node-IP}/?domain=dom1.example.com'
```

Change the domain owner

```
curl -X PUT -L
--post301
-d ''
-H 'X-Owner-Meta: ccarlin' 'http://{node-IP}/?domain=dom1.example.com'
```

Retrieve a bucket object's metadata

```
curl -I -L 'http://{node-IP}/bucket1?domain=dom1.example.com'
```

Change the bucket

```
curl -X PUT -L
--post301
-d ''
-H 'X-Owner-Meta: ccarlin' 'http://{node-IP}/bucket1?domain=dom1.example.com'
```

Note: Any *custom* metadata for domains or buckets must be included in the updates in these examples. This is performed using multiple **-H** arguments in the cURL command.

Change to anonymous ownership

Remove the **X-Owner-Meta** header or assign it a blank value with: **-H 'X-Owner-Meta: '** to change a domain or bucket to anonymous ownership so all users have full access.

Checking the Gateway Service

To troubleshoot the Gateway service:

- To find startup errors for the service, check the Gateway's server log (default: `/var/log/caringo/cloudgateway_server.log`).

Check whether the SCSP and/or S3 protocol services are running. This is an example from a server with both protocols enabled:

```
2015-09-30 11:01:10,382 INFO [main|0000000000000000] S3GatewayServlet: S3 API enabled
2015-09-30 11:01:10,382 INFO [main|0000000000000000] ScspGatewayServlet: SCSP API enabled
```

- To check whether the Gateway is responding to Management API requests, request the API version. This command is run from the Gateway and assumes the SCSP protocol is running on port 80. Adjust the port and/or interface IP address if your configuration is different:

```
curl -i http://localhost:80/_admin/manage/version
```

You should get an HTTP 200 response and a JSON body returned with that request.

Portal Upload Error

Symptom – A Content Portal upload fails, producing an error similar to this:

```
org.apache.commons.fileupload.FileUploadBase$SizeLimitExceededException. The request was rejected because its size (258196924634) exceeds the configured maximum (26355234816)
```

Cause – The Gateway's spool directory has reached its maximum capacity. The spool directory is the temporary space to support HTTP multipart MIME uploads, which are the requests that are made by uploading files into the [Content UI Overview](#).

Solution – Move the spool directory to one that has enough free space to support your largest simultaneous uploads.

In the [Gateway configuration file](#), locate the [gateway] section and edit the value for **multipartSpoolDir**, which defaults to `var/spool/cloudgateway/`.

Unexpected HTTPS to HTTP Redirects

If you are attempting access the Content Portal through HTTPS and your browser is being redirected to an HTTP link, you may need to alter your front-end load balancer settings. This issue can occur when a load balancer blindly passes back a `Location` header from the Gateway without rewriting it to match the front-end protocol of the service.

To correct for this, configure your load balancer to inject the header `X-Forwarded-Proto: https` with all SSL/TLS requests sent to the back-end pool of Gateway servers. This header instructs the Gateway to use HTTPS in any `Location` header that it sends to the browser.

Object Locking

- [Overview](#)
 - [Retention periods](#)
 - [Retention modes](#)
 - [Legal hold](#)
- [Design](#)
 - [Metadata headers related to Object Locking](#)
 - [Enabling Object Locking on a bucket](#)
 - [Locking an object at creation time](#)
 - [Managing retention on an existing object](#)
 - [Managing legal hold on an existing object](#)
 - [Combined retention and legal hold](#)
 - [Enforcing the Object Lock](#)
 - [New policy actions](#)
 - [Interactions with other functionalities](#)
 - [Content UI](#)
 - [Recursive deletes](#)
 - [Lifepoints](#)
 - [APPEND](#)
 - [Max retention config](#)
 - [Audit logging](#)

Overview

Object Locking prevents object versions from being deleted or overwritten for a fixed amount of time or indefinitely. It is applied to an object version to meet regulatory requirements that require WORM storage or to add another protection layer against object changes and deletion.

There is a strong connection between Object Locking and Versioning. Object Locking does not lock objects, but individual **object versions**. Therefore, a user can create new object versions even though the object is locked, but it is impossible to delete or change any locked version of the object.

There are two types of Object Locking:

- **Retention** - Specifies a fixed period ("retention period") during which the object version remains locked. During this retention period, the object is WORM-protected and cannot be overwritten or deleted. After the period expires, the lock goes away automatically from the object.
- **Legal hold** - Keeps the object locked until the legal hold is explicitly removed.

These two types of object locking are orthogonal, independent of each other, and can be used simultaneously.

Retention periods

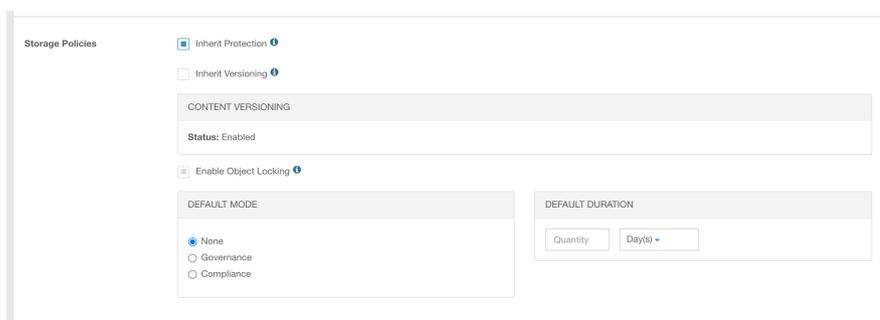
A **retention period** locks an object version for a fixed amount of time. Until that fixed amount of time has expired, one cannot delete or change the object version.

There are three different ways to set a retention period on an object version:

- Newly created objects can inherit a **default retention period** configured on the bucket.
- Explicitly set a retention period when creating a new object. This overrides the default retention period configured on the bucket if present.
- Explicitly set a retention period on an existing object version.

A bucket's default retention period specifies the duration in days or years, for which, every object version placed in the bucket is locked. While placing an object in the bucket, the Gateway calculates a retention period for the object version by adding the specified duration to the object version's creation timestamp.

An already set retention period can always be **extended**. Submit a new lock request for the object version with a retention period longer than the current period. The Gateway replaces the existing retention period with the new, longer period. Any user with permission to place an object retention period can extend a retention period.



Retention modes

There are two *retention modes* that impact what can be done with objects under retention:

- In **governance** mode, some users can be granted permission to shorten or remove a retention period if necessary.
- In **compliance** mode, any user including the admin user, cannot overwrite or delete a protected object version. When an object is locked in the

compliance mode, the retention mode cannot be changed, and the retention period cannot be shortened.

The default retention mode and retention period can be set independently at the bucket level. The retention mode always applies to the individual objects carrying it, not to the bucket or cluster as a whole.

Legal hold

A legal hold prevents an object version from being overwritten or deleted like a retention period. A legal hold does not have an associated retention period and remains in effect until removed.

Legal holds are independent of retention periods and retention modes. As long as the bucket contains an object that has Object Locking enabled, the user can place and remove legal holds regardless of whether the specified object version has a retention period set or not. Placing a legal hold on an object version does not affect the retention mode or retention period for that object version.

Version

2021-12-15 6:34:49 PM UTC ▾

Rename

Delete

Metadata

Actions ▾

Size	167.94 KB
Type	(none)
Owner	bguetzlaff@
Stored Date	2021-12-15 6:34:49 PM
X-Last-Modified-By-Meta	bguetzlaff@
Legal Hold	On
Object Lock Mode	Compliance
Object Lock Expiry	Fri, 14 Jan 2022 18:44:39 GMT
more ▾	

Important

A legal hold cannot be applied as a default at the bucket level.

Design

The Object Locking feature is implemented fully in the Gateway that relies heavily on lifepoints, a Swarm feature, which is used to prevent deletion of locked objects until a certain date has passed. It is supported by both S3 and SCSP protocols.

Despite Gateway relying on lifepoints, it remains possible for applications to impose user-defined lifepoints on objects along with object locks. The Gateway verifies correct semantics in all cases without any additional behavior needed from the application side. In the case of any conflicts between user-defined lifepoints and object locks, the object lock always wins.

Object Locking is compatible with the S3 protocol and always pertains to specific object versions. A prerequisite for applying it on a bucket is that versioning must be enabled on the bucket.

Metadata headers related to Object Locking

Object Locking uses the following headers.

- **On buckets**

```
x-object-lock-meta-status: ENABLED (absent means DISABLED)
x-object-lock-meta-default: <GOVERNANCE|COMPLIANCE>[:<duration>]
```

The duration of a bucket's default retention period is expressed as "**<integer>y**" or "**<integer>d**" for the number of years/days.

- **On objects**

```
x-object-lock-meta-mode: <GOVERNANCE|COMPLIANCE>
x-object-lock-meta-retain-until-date: <date>
x-object-lock-meta-legal-hold: ON (absent means OFF)
x-object-lock-meta-original-lifepoints: <original lifepoints>
lifepoint: [<date>] deletable=no (for retention period)
lifepoint: [] deletable=no (for legal hold)
```

The above headers are listed using the SCSP names and the corresponding S3 names start with `x-amz-*`. The SCSP headers are effectively stored with objects. The S3 names are mapped onto the SCSP counterparts and back on the fly.

Internal to Gateway, all header values are treated case insensitive. Dates are in **rfc1123** format, e.g., "Wed, 12 Dec 2016 15:59:02 GMT". For S3, these are translated in to **ISO8601** format.

- The `x-object-lock-meta-retain-until-date` header applies to retention periods and specifies the end date of the retention period.
- The `x-object-lock-meta-legal-hold` header applies to the legal hold.

The `x-object-lock-meta-original-lifepoints` header stores the complete set of user-defined delete/deletable lifepoint headers found on the object when the retention period/legal hold is applied. The original delete/deletable lifepoint headers are removed.

ⓘ The term "lifepoint" implicitly means the delete/deletable lifepoints. All other types of lifepoints are not affected by object locks.

The lock lifepoint protects the object against deletion in Swarm, so be it through user requests or built-in functionalities like HP or bucket policies. The lock lifepoint is computed as follows:

- If the object is locked with a retention period, then the lock lifepoint end date matches the end date of the retention period. For legal hold, the lock lifepoint has no end date.

- Go over the list of original lifepoints and append those whose end date is later than the one from the lock lifepoint. In the case of the legal hold, there is no such end date so none of the original lifepoints get appended.

The purpose of storing the set of original lifepoints is to allow later modifications/removal of the object lock to recompute/reinstate the original lifepoints as they are before the object locking.

The purpose of appending the "later lifepoints" to the lock lifepoint is to allow Swarm to act on them as it normally does once the lock lifepoint has expired naturally, without any gateway intervention. For legal hold, there must always be a gateway intervention to remove the lock, so the original lifepoints are reinstated at that time.

Enabling Object Locking on a bucket

Before locking any objects, verify Object Locking is enabled on the bucket. S3 allows enabling Object Locking on new buckets carrying no objects, but Gateway does not impose this restriction. The user must have `PutBucketObjectLocking` permission to enable/disable Object Locking on a bucket. The user must have `GetBucketObjectLocking` permission to query the current Object Locking status.

The request to enable Object Locking can fail with the following errors:

Error code	Definition
412	When the bucket does not have versioning enabled. It does not matter if the versioning was enabled on the bucket itself, or whether it was inherited from a cluster or domain level.
403	When the user does not have PutBucketObjectLocking permission.

Enabling Object Locking on a bucket comes down to storing the `x-object-lock-meta-status` and optionally `x-object-lock-meta-default` headers on the bucket context object. Using S3, one enables/inspects Object Locking config on a bucket using the following calls:

- https://docs.aws.amazon.com/AmazonS3/latest/API/API_PutObjectLockConfiguration.html
- https://docs.aws.amazon.com/AmazonS3/latest/API/API_GetObjectLockConfiguration.html

Using SCSP, one enables Object Locking on a bucket as follows:

```
PUT /<bucket>?objectlock=<defaultmode> [ :<defaultperiod> ] where;
```

- The default mode is either **governance** or **compliance**.
- The optional default period is a number of years (y) or days (d), e.g., 1y or 20d.

In this call, the user can omit either default mode, default duration, or both. The defaults can be modified or removed at any time via additional **PUT** command.

 In a deviation from S3, the Gateway always uses the MAXIMUM of either the bucket default retention duration, or duration specified in a per-object request.

Use `GET /<bucket>?objectlock` to query object locking status of a bucket. This returns the following response headers:

```
x-object-lock-meta -status: ENABLED
x-object-lock-meta-default: <GOVERNANCE|COMPLIANCE>[:<duration>]
```

And the response body says:

Object locking is enabled on bucket <bucket> with default mode <mode> [and default duration <duration>]

No response headers are present if the bucket does not have object locking enabled and the response body says:

Object locking disabled

Object Locking cannot be disabled once enabled on a bucket. It is possible to remove the lock defaults and provide lock headers on writing new objects to force a lock to allow writing unlocked objects in a bucket with Object Locking enabled.

Locking an object at creation time

The client adds the following headers in the S3 PutObject/SCSP POST request to create a new object with an immediate retention period in effect;

```
x-object-lock-meta-mode: <GOVERNANCE | COMPLIANCE>
x-object-lock-meta-retain-until-date: <date>
```

This takes precedence over the default bucket retention mode and duration if present.

- The Gateway looks for corresponding defaults at the bucket level if any one of these two headers are omitted from the request.
 - It takes the corresponding values from there if found.
 - The request fails with an HTTP 400 error as both are needed for a successful retention lock if either mode or retain-until-date is still missing.
- The object is written as a normal unlocked object, despite being written to a bucket that has Object Locking enabled if both headers are omitted from the request and there is no default set at the bucket level. The client can request an immediate legal hold by specifying the header:

```
x-object-lock-meta-legal-hold: ON
```

Use of these headers requires that the user has `PutObjectRetention/PutObjectLegalHold` permission; otherwise the request fails with an HTTP 403 error. The request fails with an HTTP 412 error if the bucket does not have Object Locking enabled, .

The Gateway forwards these headers when creating a new object on Swarm, and also creates a lock lifepoint instructing Swarm to not delete the object before the retention period expires.

Lock Mode	Lock Lifepoint	Remarks
Retention Period	lifepoint: [<date>] deletable=no, <later lifepoints>	Lock lifepoints includes the subset of the original lifepoints that had a later end date than the retention period.
Legal Hold	lifepoint: [] deletable=no	

Finally, the original lifepoint headers are preserved in `x-object-lock-meta-original-lifepoints: <original lifepoints>`

Managing retention on an existing object

Enabling/disabling retention on an object requires that the user has `PutObjectRetention` permission. The user must have `GetObjectRetention` permission to query the current retention status. The client must explicitly specify the `versionId` of the object version to lock.

The Gateway then creates a new variant of that version on which it stores the extra Object Locking headers. This variant is protected by the retention period. For the client, there is no distinction between the variant and the original object version; the following headers are added or changed:

```
x-object-lock-meta-mode: <GOVERNANCE | COMPLIANCE>
x-object-lock-meta-retain-until-date: <date>
x-object-lock-meta-original-lifepoints: <original lifepoints>
lifepoint: [<date>] deletable=no, <later lifepoints>
```

Introducing or extending a retention period is always possible, but there are restrictions to shortening/removing a retention period on an object that is already under the retention:

- In compliance mode, this is not permitted
- In governance mode, it requires that the user has a special permission `BypassGovernanceRetention`.

An S3 request must explicitly include `x-amz-bypass-governance-retention:true` as a request header with any request that requires overriding governance mode. Using S3, one enables/inspects a retention period on an object using the following calls:

- https://docs.aws.amazon.com/AmazonS3/latest/API/API_PutObjectRetention.html
- https://docs.aws.amazon.com/AmazonS3/latest/API/API_GetObjectRetention.html

Using SCSP, one enables/inspects a retention period on an object using the following calls:

Call type	Purpose
<pre>PUT /<bucket>/<object>? version=<uuid>&objectlock=governance:<untildate></pre>	To put a governance lock onto an object and specify both lock mode and duration. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> i This overrides any defaults configured on the bucket. </div>
<pre>PUT /<bucket>/<object>? version=<uuid>&objectlock=compliance</pre>	To put a compliance lock onto an object and inherit the default duration from the bucket.
<pre>PUT /<bucket>/<object>?version=<uuid>&objectlock</pre>	To inherit the default object lock mode and duration on the bucket.
<pre>DELETE /<bucket>/<object>? version=<uuid>&objectlock=<mode></pre>	To remove an object lock from an object, assuming the user has the proper permissions. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> i The mode is always <i>governance</i> as the compliance mode object lock cannot be removed. </div>

Note

In addition the user must have `BypassGovernanceRetention` permission to carry out this action and the request must carry the `x-object-lock-meta-bypass-governance:true` header.

Use `GET /<bucket>/<object>?version=<uuid>&objectlock` to query the current object lock status, and the response carries the following headers:

```
x-object-lock-meta-mode: <GOVERNANCE | COMPLIANCE>
x-object-lock-meta-retain-until: <date>
```

And the response body says:

Object is locked in *<mode>* mode until *<date>*

When called on an object that is not under retention, none of the headers are present and the response body says:

Object is not locked

Both S3 and SCSP allow retrieving Object Lock information using the regular object HEAD and GET requests. Assuming that the user has `GetObjectRetention` permission, the information is returned in the form of the above response headers. The response body is not affected.

Managing legal hold on an existing object

Enabling/disabling legal hold requires `PutObjectLegalHold` permission for the user. To query the current legal hold status, the user needs `GetObjectLegalHold` permission.

```
x-object-lock-meta-legal-hold: ON (absent means OFF)
x-object-lock-meta-original-lifepoints: <original lifepoints>
lifepoint: [] deletable=no
```

Using S3, the user can enable/inspect legal hold using the following calls:

- https://docs.aws.amazon.com/AmazonS3/latest/API/API_PutObjectLegalHold.html
- https://docs.aws.amazon.com/AmazonS3/latest/API/API_GetObjectLegalHold.html

Using SCSP, the user can enable/inspect legal hold on an object using the following calls:

Call Type	Purpose
PUT /<bucket>/<object>?version=<uuid>&objectlock=legal-hold	To put a legal hold onto an object
DELETE /<bucket>/<object>?version=<uuid>&objectlock=legal-hold	To remove legal hold from an object. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p> This reinstates any original lifepoints by moving them from the <code>x-object-lock-meta-original-lifepoints</code> header to the <code>lifepoint</code> headers.</p> </div>
GET /<bucket>/<object>?version=<uuid>&objectlock	To query an object's legal hold status.

The response carries the following headers:

x-object-lock-meta-legal-hold: on.

And the response body says:

Object is locked in legal hold

When it is called on an object that is neither under the legal hold nor retention, the header is not present, and the response body says:

Object is not locked

Both S3 and SCSP allow retrieving legal hold information using regular object HEAD and GET requests. Assuming the user has `GetObjectLegalHold` permission, the information is returned in the form of the above response headers. The response body is not affected.

Combined retention and legal hold

An object can fall under both retention mode and legal hold at the same time. In the SCSP protocol, querying and deleting such combined locks are handled via a uniform *GET* and *DELETE* API (as opposed to S3 which has separate APIs for querying/deleting retention and legal hold).

For query, use `GET /<bucket>/<object>?version=<uuid>&objectlock[=<locktype>]`

When querying the object lock status without specifying the lock type, response headers for both retention and the legal hold display, and the response body contains both status texts, separated by a newline.

Important

The user needs both **GetObjectRetention** and **GetObjectLegalHold** permissions for this request.

One can also query lock status for one specific lock type, being either legal hold or retention. The corresponding permission is required.

To delete, use `DELETE /<bucket>/<object>?version=<uuid>&objectlock=<locktype>`. Using SCSP one can remove either the retention or legal hold using `DELETE` and can specify the appropriate query argument `objectlock=<locktype>`, where *locktype* is *legal hold* or *retention*.

Enforcing the Object Lock

It is worth repeating that Amazon S3's definition of Object Locking locks object versions, and not objects. It is perfectly possible to overwrite an object that is locked, and now, the overwritten version continues to exist and is protected from changes or deletion.

Moreover, object versions are immutable in Swarm; any object version is protected from modification by design. The exception to this is deletion, which the lock lifepoints now protect against:

- The users' attempts to delete
- Any automated delete attempts by Swarm.

Swarm rejects any delete requests for indelible objects with a 403 error. An S3 client can delete the delete marker, making the previous version visible again in the Portal. An upcoming Portal release allows visibility of delete markers and version history in a bucket listing.

For SCSP, this is an update to [config.cfg](#) to pick the desired behavior which is either 'fail deletes the locked objects with a 403 error' or 'mimic the S3 behavior'.

```
[object_locking]
scspDeleteUsesS3Logic=true
```

New policy actions

The following new policy actions related to Object Locking are introduced:

Policy actions	Definition
PutBucketObjectLocking	To enable/disable object locking on a bucket
GetBucketObjectLocking	To query bucket object locking status
PutObjectRetention	To put or extend object retention
GetObjectRetention	To query an object retention

BypassGovernanceRetention	To shorten/remove retention in the governance mode
PutObjectLegalHold	To put/remove a legal hold
GetObjectLegalHold	To query a legal hold

Interactions with other functionalities

Content UI

The Content Portal also supports Object Locking.

- There are different icons based on each Object Locking state and default of the bucket in the bucket listing view.
- The object versions are locked at the bucket level.
- Both retention and legal hold can be applied on a single object version if necessary.

Recursive deletes

To avoid the conflict of recursive deletes attempting to remove a locked object, Gateway first checks if there are any objects under retention or legal hold and refuses the recursive delete if so. The request fails with an HTTP 412 error if the recursive delete cannot be performed due to the Object Locking.

While the recursive delete of a domain or bucket does not immediately result in deletion of locked object versions, instead this makes it less practical to find and access. Significantly, it defeats the built-in safety checks that prevent versioning from getting disabled, which results in the deletion of the locked object versions. For that reason, the Gateway requires an additional config setting in the **gateway.cfg**;

```
[object_locking]
allowRecursiveDeleteBypass=true
```

A warning is logged if this setting is present. The header into the recursive delete request is `X-Object-Lock-Meta-Bypass-Recursive-Delete-Check: true`. When encountering this header, the Gateway skips the aforementioned checks and allows the recursive delete to proceed.

Lifepoints

The original set of user-defined lifepoints is preserved in a separate header and can be reinstated when the object lock is removed. This applies to deletable/delete lifepoints, all other types of lifepoints are left as is.

The Gateway adds a `deletable=no` lock lifepoint to protect locked objects from inadvertent deletion. In the case of the retention period, the lock lifepoint has the same end date as the retention period. The lock lifepoint includes the subset of user-defined lifepoints with a later end date than the retention period. This allows those lifepoints to automatically resume taking effect as soon as the retention period expires.

In the case of the legal hold, the lock lifepoint has no end date, and no user-defined lifepoints are included in it.

APPEND

SCSP APPEND creates a new version when versioning is enabled. Any lock either retention or legal hold, which is applied to the object version, is applied to the new version created by the APPEND operation.

Max retention config

S3 allows defining a "max-retention-duration" limit in the policy. The Gateway has the new configuration option to offer a similar capability to S3. Using a single new config flag to approximate this functionality:

```
[object_locking]
retentionMaxYears=100
```

The default limit value is 100 years if unspecified. It is assumed a year is 365 days when performing conversions between numbers of days and years, .

In the SCSP/S3 APIs, any user-specified value exceeding the limit is silently capped to the limit.

Audit logging

Object Locking operations are audit logged. Since object locks can also be requested as part of the object PUT/POST/COPY requests, the Gateway tags the request's audit log line with additional object lock information, rather than inserting new log lines.

The tags are appended to the audit log line, enclosed in '[]' brackets. Multiple tags (both legal hold and retention) are separated by a comma. Object locking tags are always prefixed with string **OBJLCK**. This keeps a door open for other kinds of tags in the future.

Object Locking Operations	Audit Log Line
Enabling retention on a bucket and setting defaults if provided.	<audit log line> [OBJLCK:ENABLE:<mode>:<duration>]
Setting/removing retention on an object.	<audit log line> [OBJLCK:RETENTION:<mode>:<retainUntil>] <audit log line> [OBJLCK:RETENTION:NONE]
Setting/removing legal hold on an object.	<audit log line> [OBJLCK:LEGALHOLD:ON] <audit log line> [OBJLCK:LEGALHOLD:OFF]

Swarm Content UI

The Content UI is Swarm's cloud service for browser-based content management.

- [Configuring Domains](#)
- [Setting Identity Management](#)
- [Configuring Buckets](#)
- [Using Virtual Folders](#)
- [Usage Reports](#)
- [Setting Storage Policies](#)
- [Video Clipping for Partial File Restore](#)
- [Configuring Tenants](#)
- [Metadata Encoding](#)
- [Setting Remote Synchronous Write \(RSW\)](#)
- [Editing Names, Metadata, and Versions](#)
- [Using the Content UI](#)
- [Search Collections](#)
- [Setting Permissions](#)
- [Uploading Files](#)
- [Setting Quotas](#)
- [Setting Tokens](#)
- [Content UI Overview](#)
- [Downloading Content](#)
- [Object Locking Content Portal](#)
- [Swarm Hybrid Cloud](#)

Configuring Domains

- [Domain Essentials](#)
- [Domain Properties](#)
- [Permissions](#)
- [Tokens](#)

Domain Essentials

Within a tenant, a domain is the primary entity for dividing and controlling access and resources. Domains have these essential features:

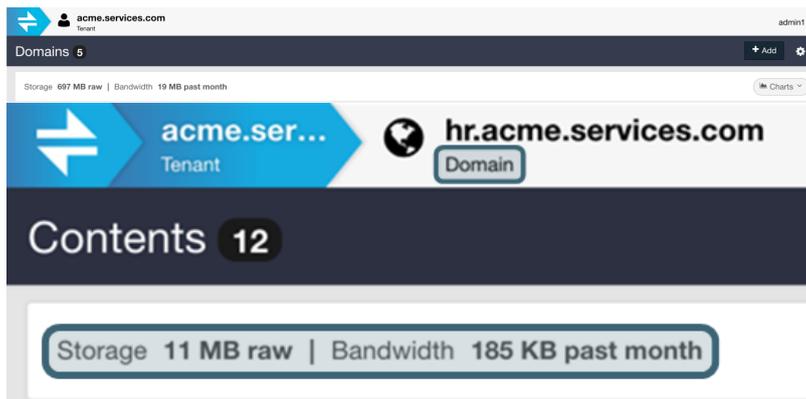
- **Ownership.** Each domain owns one or more buckets.
- **Access control.** Domains can define separate identity management system so the users and groups within them are separated from those in other domains.
- **Delegation.** Domain administrators can create and access storage domains and they can delegate management duties for the storage domains they create.
- **Content.** The domain itself stores *buckets* for named objects for end-user data and *collections* (stored searches).

Unnamed objects

Unnamed objects written directly to the domain are represented by a system-defined **Content IDs** bucket part of each domain.

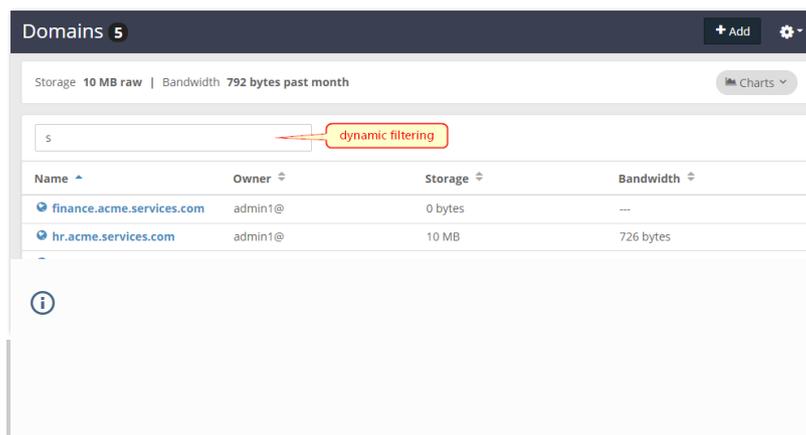
See the [Naming Rules for Swarm](#) for domains.

Domain Usage – The **Storage Used** chart displays the current current size of the storage footprint used by all domains, inclusive of all versions, replicas, and erasure-coded segments when viewing *all* domains in a given tenant. The **Bandwidth Used** chart displays the total bandwidth (bytes in and bytes out) used by each domain over a rolling 30-day window. See [Usage Reports](#).



A domain reports the usage at the very top, along with the total bucket and collection count when opening up a domain:

Dynamic Filtering – Narrow the listing by entering a string in the **Filter** box, which filters by **Name** if a large number of domains exist.



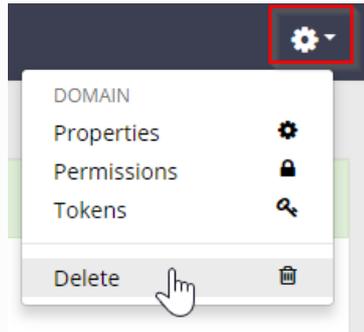
Default Items – Every domain is created with standard built-in items to help manage the contents:

- A special system-generated bucket for unnamed objects (**Content IDs**),
- A set of default search collections, for commonly needed views in to the content, by age and type

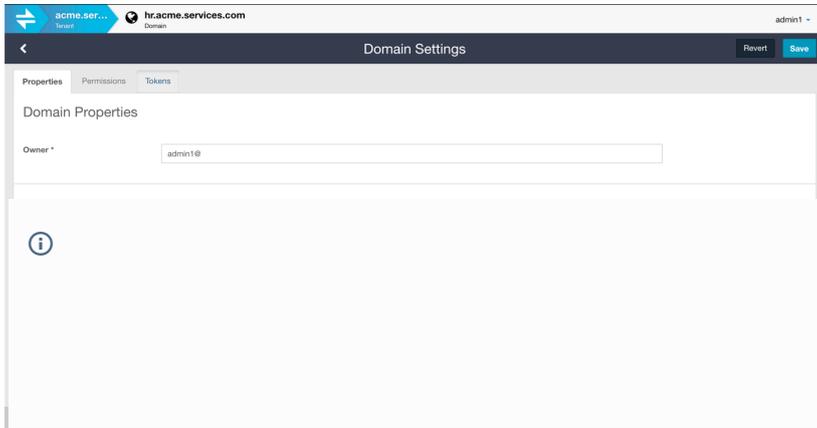
Delete Domain

The **Delete** command deletes the domain and *all* buckets and uploaded contents and any saved collections for the domain.

Warning: This command cannot be undone, so proceed with caution.



Domain Properties



Owner

Every domain needs an owner with access to and ultimate authority over. Create a domain for another to manage as a root or tenant admin.

Note

Ownership defaults to the specific administrator who created the domain, but the owner does not need to be a root or tenant administrator.

Change the owner when creating a context for someone else to manage. One does not want to own or be responsible for managing the data in the domain when creating a domain for a client.

Quotas

Quotas can be set to determine how much storage and/or network bandwidth the domain is permitted to consume.

See [Setting Quotas](#).

Storage Policies

Storage policies control how this domain's objects are protected (using replication and/or erasure coding) and whether versioning is in use. The domain inherits the storage policies in force for the cluster by default.

Specify custom policies if inheriting these policies is disabled, but these custom policies are subject to what is allowed and in force in the cluster. A warning icon and message alerts to the situation if opting for something being overridden by a higher policy.

See [Setting Storage Policies](#).

Identity Management

The IDSYS objects define the identity management systems controlling the domain's users:

- User and group information
- The authentication system

See [Setting Identity Management](#) and [Gateway Identity System](#).

Permissions

Permissions are determined by the access control policy, which are the rules granting (or denying) users and groups the ability to perform specific actions.

See [Setting Permissions](#).

Tokens

See [Setting Tokens](#).

System Domain and Legacy Mode for Gateway

- [System domain vs. Default domain](#)
- [API and UI](#)
- [Setting Up Access Permissions](#)
- [Configuring a Gateway as a System Domain-only Gateway \(Legacy Mode\)](#)

Starting with Gateway 7.3 and Content UI 7.3, the concept of a System domain has been introduced to provide legacy SCSP clients with the ability to access [unnamed](#) objects stored outside of all storage domains. The System domain feature allows taking advantage of Swarm's modern features such as metadata searching for unnamed and untenanted objects in a cluster. It provides better access control policy management and integration via the UI.



System domain vs. Default domain

System domain is not the same as a default domain. For more information on Default domain, see [Guidelines for managing Domains](#).

With the System domain, the choices for connecting legacy SCSP clients with the storage are:

1. direct network connection to all object storage nodes,
2. through legacy SCSPproxy package, or
3. through gateway running in legacy mode.

Direct network connection and SCSPproxy with legacy application clients:

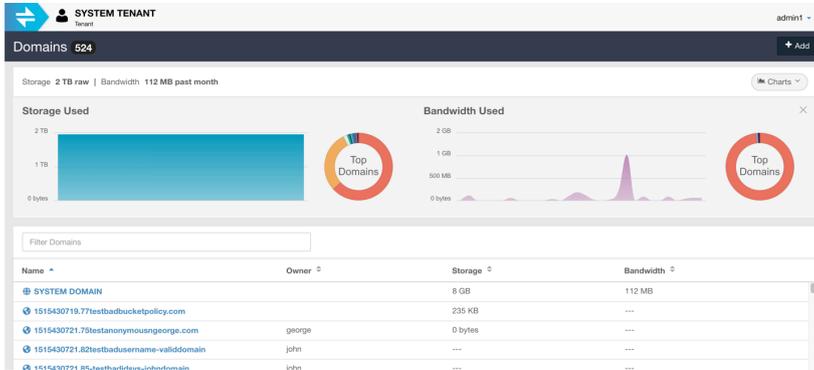
- continue to work in existing deployment without code modifications
- can use legacy HTTP digest auth/auth mechanism with storage nodes
- storage-in-use metering is tracked by gateway
- bandwidth metering is not tracked by gateway
- no audit log tracking by gateway
- can interfere with tenanted content within storage domains – depends on specific application

Legacy application clients connecting through gateway:

- continue to work without changing application code logic (except legacy auth/auth)
- cannot use legacy HTTP digest auth/auth mechanism
- storage-in-use and bandwidth metering is tracked by gateway
- audit logging for all access
- access control using gateway's policy mechanism
- assured isolation from content within other storage domains

API and UI

The System domain is considered a child of the System tenant and is represented as a domain called "System" within the System tenant, both in the UI listing and in the Management API ("_system"). Metrics for the System domain roll up into the System tenant, together with metrics for all untenanted domains.

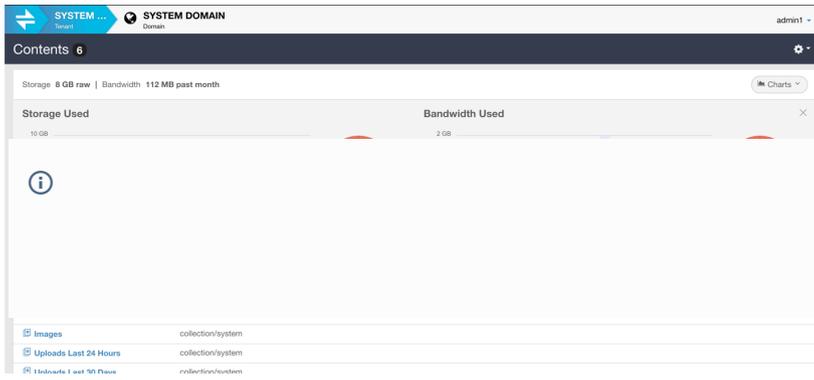


Buckets cannot be created in the System domain, but it presents the Content IDs pseudo-bucket. Upload to Content IDs the same way as to any other domain.

System domain also supports Collections.

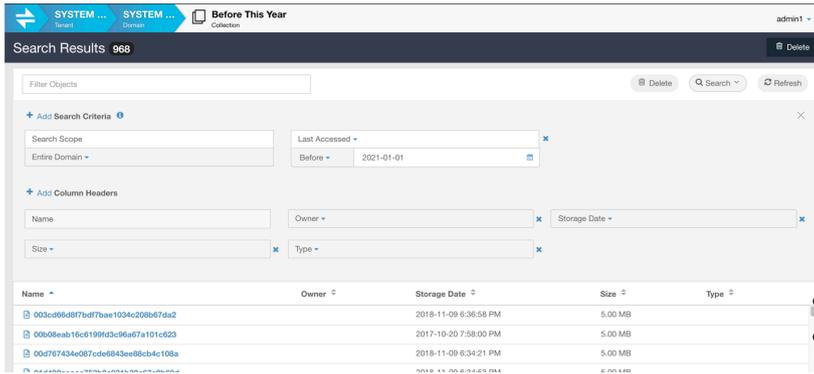
Setting Up Access Permissions

The System domain has no owner and no one can be assigned to be the owner, so there is no default access policy for it. System domain management only allows setting [IDSYS](#) and policy based access. Access to content in the System domain must be granted through the root and/or System domain-specific policies.



No user is able to perform SCSP operations with content in the System domain if no policy is added and no root policy exists granting access to the System domain.

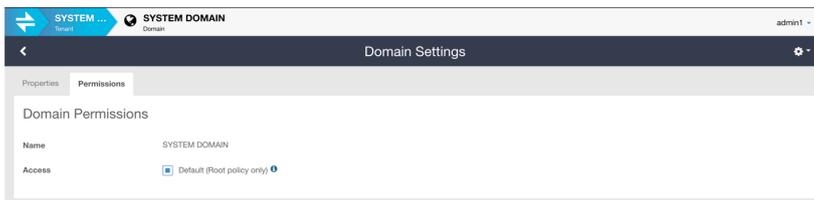
[Authentication tokens](#) are not supported for the System domain in the UI.



Configuring a Gateway as a System Domain-only Gateway (Legacy Mode)

Gateway can be configured to work in one of the following modes:

- **Normal mode** with tenanted named and unnamed objects
- **Legacy mode** with unnamed untenanted objects only. (new with v7.3)



This is configured using the following setting. The default value is 'false' and the gateway runs in normal mode if unset.

[gateway]
legacyOnlyMode = true/false

Legacy mode allows configuring a gateway as a System domain-only gateway for use by legacy SCSP clients so unnamed objects in the System domain can be accessed. Gateway disregards a client's specification domain and communicates solely to the System domain in the back-

end storage cluster when operating in this mode.

Content UI is only available through normal mode gateways and attempting to use the UI through a legacy-only mode gateway returns the following message in a browser:

This gateway is running in legacy mode. UI requests are not supported.

 Attempting to use modern clients using tenanted objects within storage domains or named objects within buckets with a gateway configured in legacy mode is a misconfiguration. These clients need to use a separate gateway configured for normal mode operations.

Setting Identity Management

- [Defining a New IDSYS](#)
 - [Important](#)
 - [Tip](#)
 - [Testing the Identity Configuration](#)
 - [Best practice](#)
- [Defining SSO \(SAML\)](#)

The **Identity Management** section of **Properties** allows defining an overriding identity management system (IDSYS) that authenticates the users of the specific tenant or domain.

Authenticating at this more granular level enables enforcing context-specific control, such as to:

- Authenticating client's users so they can be granted access within the customer's designated tenant area only
- Authenticate a managed group of users for a specific domain, such as for a business function, division, or region

See [Gateway Identity System](#).

Defining a New IDSYS

By default, every tenant inherits the *root* configuration, and every domain inherits from its parent tenant. Create a custom configuration by disabling (unchecking) **Inherit**.

Important

Once **Inherit** is disabled, all connection to IDSYS changes occurring at the higher levels is ended until **Inherit** is enabled again.

From the **Templates** drop-down list, copy existing definitions to alter. Changes do not affect the originals.

Tip

Select **Revert** to restore the last saved script if enabling **Inherit** removed the initial script.

Scripts are validated in real time:



Testing the Identity Configuration

To test the identity management configuration, click **Test**, enter a user name and password pair, and then click **Test**.

Important

Best practice

Test invalid as well as valid user name and password pairs.

Defining SSO (SAML)

With Gateway 7.1 and Content UI 7.0 and higher, enable single sign-on for tenants and/or domains to access the Content UI through a third-party identity provider. See [Enabling SSO with SAML](#). (v7.0)

The starter SAML script populates in the editing box when **SAML** is selected from **Templates**. Once the `entity` field is assigned and is updated with values from an identity provider (such as Google), the **Identity Provider (IdP) Resources** below the box has meaningful values that help complete the SSO setup with IdP:

```
1 "saml": {
2   "cookieName": "token",
3   "tokenPath": "/.TOKEN/",
4   "entity": "Your root organization Service Provider entity",
5   "idpEntityId": "The Identity Provider entity ID.",
6   "idpSsoUrl": "The Identity Provider Single-Sign-On (SSO) URL, if applicable.",
7   "idpSloUrl": "The Identity Provider Single-Log-Out (SLO) URL, if applicable.",
8   "groupAttrName": "User attribute name where group information can be found, if applicable.",
9   "idpCert": ""
10 }
11 }
12 }
```

Open the link, **Service Provider Attributes**. Open the **Service Provider Metadata XML** file if the IdP cannot import.

Information you will need to provide to your Identity Provider (IdP).
See documentation for details.

Copy to Clipboard

Entity ID: Your root organization Service Provider entity ID/acme.services.com/hr.acs
Assertion Consumer Service (ACS) URL: http://cloudscaler6.8081/_admin/saml/login
ACS Request Method: POST
Single Log-Out (SLO) Service URL: http://cloudscaler6.8081/_admin/saml/logout/acs

Close

Configuring Buckets

- [Bucket Essentials](#)
- [Naming Buckets](#)
- [Bucket Properties](#)
- [Permissions](#)

Bucket Essentials

Within a domain, a bucket is the primary entity for managing uploaded content. Buckets have these essential features:

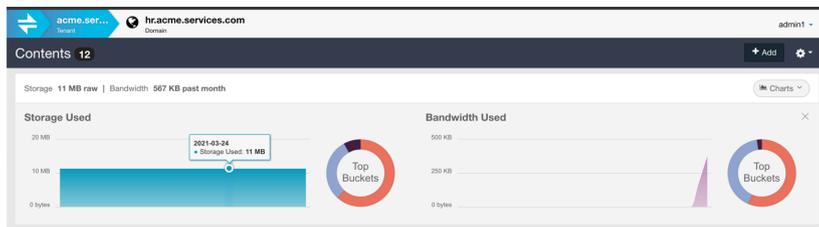
- **Ownership.** Each bucket has an owner.
- **Access control.** Buckets can define separate permissions so users and groups within them are separated from those in other buckets.
- **Content.** The bucket itself stores end-user data as named objects.

Unnamed objects

Unnamed objects cannot be written to buckets, but they can be written directly to the domain itself. Unnamed objects are contained in the system-controlled, read-only **Content IDs** bucket part of each domain.

See the [Naming Rules for Swarm](#) for buckets and [Bucket Restrictions in Amazon S3](#).

Bucket Usage – The **Storage** column displays the current size of the storage footprint used by all buckets, inclusive of all versions, replicas, and erasure-coded segments when viewing *all* buckets in a given domain. The **Bandwidth** column displays the total bandwidth (both bytes in and bytes out) used by all buckets over a rolling 30 day window. See [Usage Reports](#).



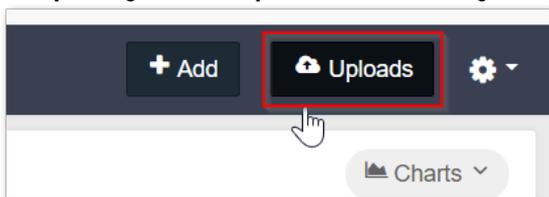
A bucket reports its *own* usage at the top of the page, along with its total object count:



Dynam

– Narrow the listing by entering a string in the **Filter** box, which filters by **Name** if a large number of buckets and/or collections exist.

File Uploading – Use the **Uploads** icon to the far right of the listed buckets to initiate uploads from the local file system.



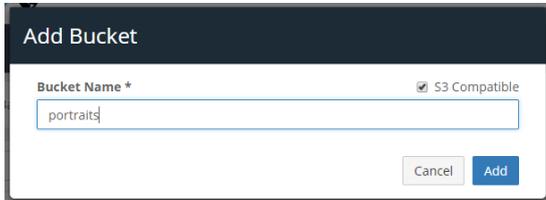
Delete Bucket

The **Settings > Delete Bucket** command deletes the bucket and *all* uploaded contents. This command cannot be undone, so proceed with caution.

Deleted objects may continue to appear in Collection listings for a period of time after they are deleted, but they are no longer accessible in the cluster.

Naming Buckets

Provide a name to add a bucket. The **Add** button becomes active for selection when the name is validated:



The screenshot shows a dark-themed dialog box titled "Add Bucket". It contains a text input field labeled "Bucket Name *" with the text "portraits" entered. To the right of the input field is a checked checkbox labeled "S3 Compatible". Below the input field are two buttons: "Cancel" and "Add". The "Add" button is highlighted in blue, indicating it is active.

S3 compatible – The bucket name needs to use lowercase alphanumeric characters and stay within 3 to 63 characters in length. The name is validated with dynamic feedback while typing:



The screenshot shows the same "Add Bucket" dialog box, but the text input field is empty. The "Add" button is now greyed out, indicating it is inactive due to the lack of a valid name.

See [Bucket Restrictions in Amazon S3](#).

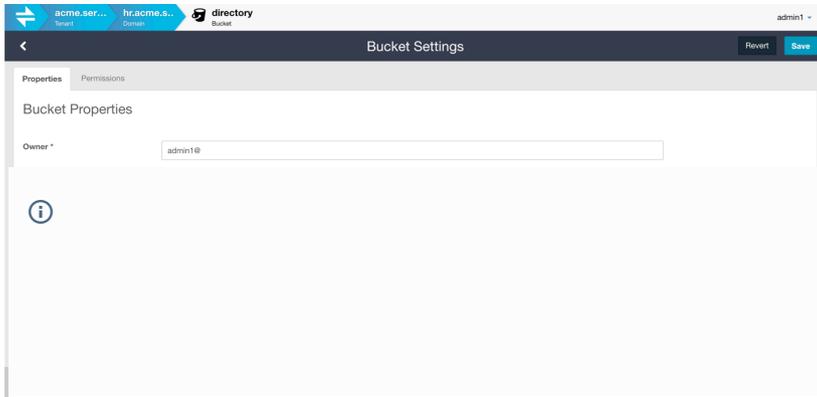
ⓘ

compatibility of bucket names is improved with any future application integrations needed.

Tip

By following S3 compatibility restrictions in naming, general

Bucket Properties



Owner

Every bucket must have an owner, who has access to and authority over its entirety.

Note

Ownership defaults to the person who created the bucket, but the owner does not need to be a domain administrator.

Change the owner when creating a context for someone else to manage. Do not own or be responsible for managing the data in the bucket if creating a bucket for a client to use with an application.

Storage Policies

Storage policies control how this bucket's objects are protected (via replication and/or erasure coding) and whether they are versioned. By default, the bucket inherits the storage policies in force for the cluster and the domain.

Specify custom policies, but these custom policies are subject to what is allowed and in force in the cluster and the domain if inheriting these policies is disabled. A warning icon and message alerts to the situation if opting for something being overridden by a higher policy.

See [Setting Storage Policies](#).

Permissions

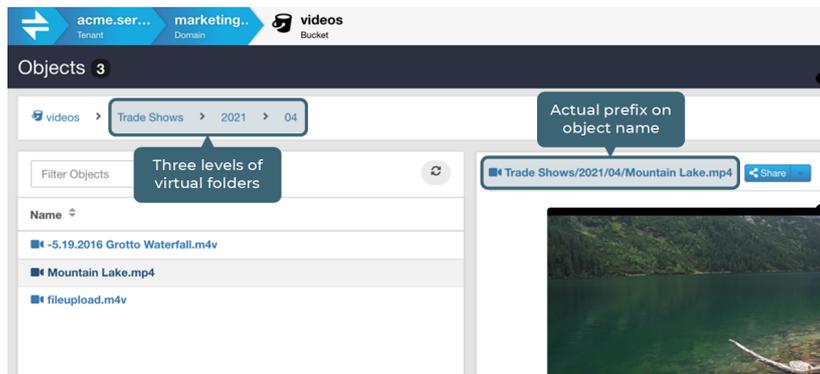
Permissions are determined by the access control policy, which are the rules granting (or denying) users and groups the ability to perform specific actions.

See [Setting Permissions](#) and [Gateway Access Control Policies](#).

Using Virtual Folders

- [Navigating Virtual Folders](#)
 - [Folder view](#)
 - [Flat view](#)
- [Adding Virtual Folders](#)
 - [Temporary](#)
 - [Important](#)
 - [Permanent](#)
- [Deleting Virtual Folders](#)
 - [Recursive delete](#)
 - [Non-recursive delete](#)
 - [Tip](#)

The Content UI presents a dynamic folder hierarchy for browsing and uploading content within a bucket. This hierarchy is based on the prefixes stored on the object names (such as `folder/subfolder/object.xml`). This folder simulation is similar to the listing behavior of other visual clients used with Swarm, such as SwarmFS and S3 Browser. (v7.0)



These folders offer three key benefits:

Prefix filtering – By parsing object prefixes into hierarchical folders in real time, the Content UI provides users a fast and intuitive way to view and manage content in a bucket, automatically.

Empty folders – The Content UI allows *creation* and persistence of new, empty folders ready to receive files. This allows planning and setting up organizing structures ahead of time, to guide content uploaders to use the organization. By having users upload directly to folders, enforce a content architecture and avoid the risk they perform bulk uploads using a malformed prefix.

- **Recursive deletes** – Even more powerfully, the Content UI allows deleting virtual folders, which recursively deletes objects *and subfolders* they contain. Users are warned about the impact and are prompted to verify all folder deletes.

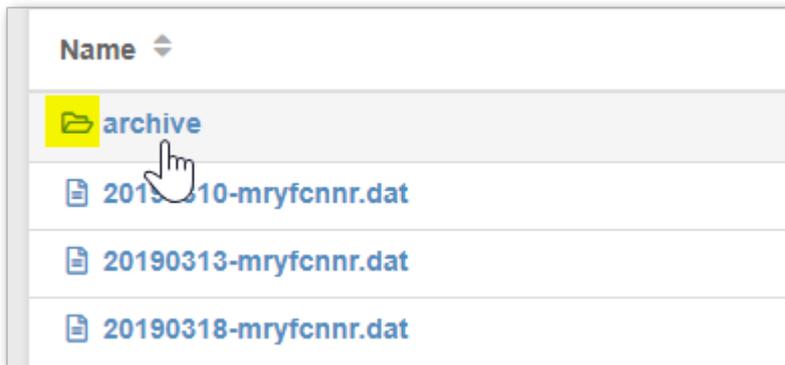
Navigating Virtual Folders

The Content UI automatically parses object names into a folder hierarchy when opening a bucket to view the contents.

Folder view

This is the automatic view of the bucket contents; one folder level at a time is visible. Although S3 includes the folder *within* the listing, the Content UI does not, so as to match the behavior of file systems.

- To walk *down* the hierarchy of folder levels, click on a folder name in the object listing, which opens that folder level.

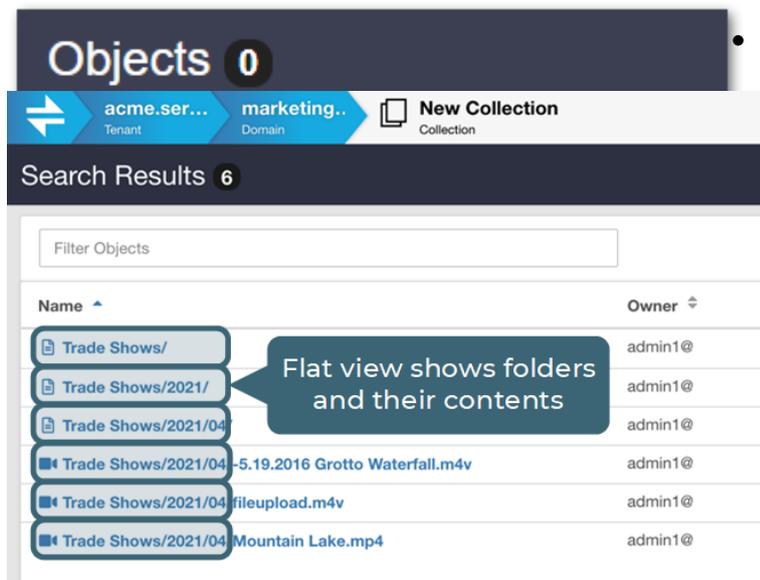


- To walk *up* the hierarchy of folder levels, click on a folder name in the breadcrumb trail, which opens that folder level.
- To see the full name of a given object, click the name to open the Detail pane:

Flat view

The ability exists to see the flattened view of bucket contents, without virtual folders.

- To see the flattened view, start a Collection (which is a saved Search):



Adding Virtual Folders

The Content UI displays virtual folders in two ways – one temporary and one permanent.

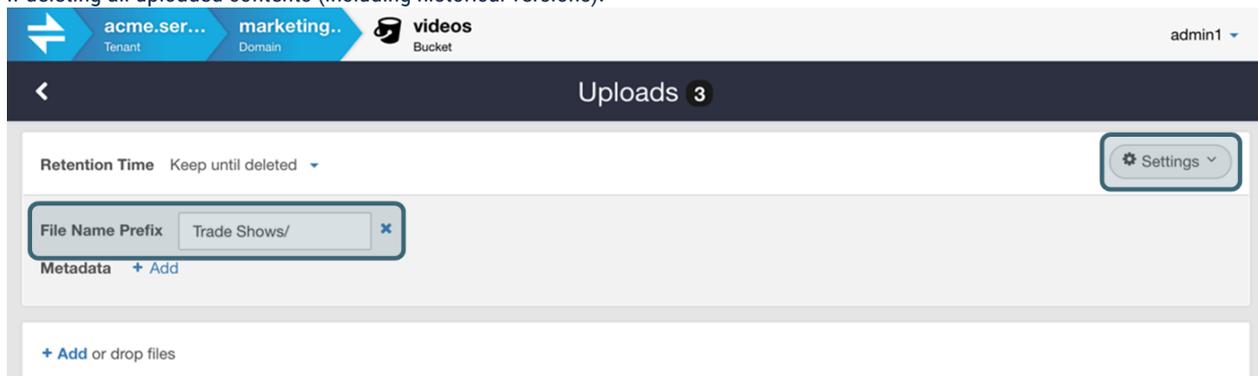
Temporary

Create temporary folders when content is uploaded with a **File Name Prefix**.

i **Important**

With prefixes, naming is important: do not begin with a slash, but always end with a slash: `myfolder/`

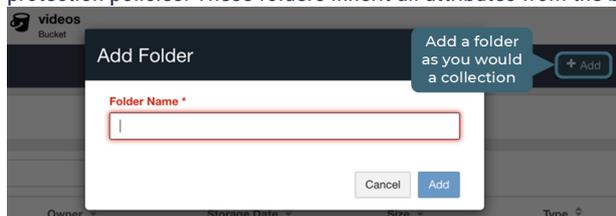
- The uploaded content appears in the virtual folder because it is synthesized dynamically; the folder no longer appears in the bucket listing if deleting all uploaded contents (including historical versions).



Permanent

To create a persistent folder even when empty, open a bucket and select **+Add > Add Folder**.

- This action creates a durable placeholder folder preserved in the bucket until it is explicitly deleted.
- These folders are virtual and are *not* full contexts (like domains and buckets) carrying customizable metadata, permissions, and content protection policies. These folders inherit all attributes from the bucket.



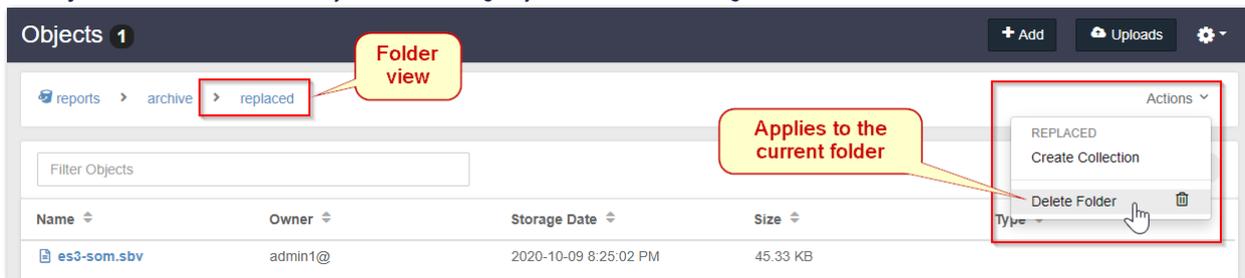
Deleting Virtual Folders

The Content UI allows deleting permanent virtual folders in two ways – one recursively and one non-recursively:

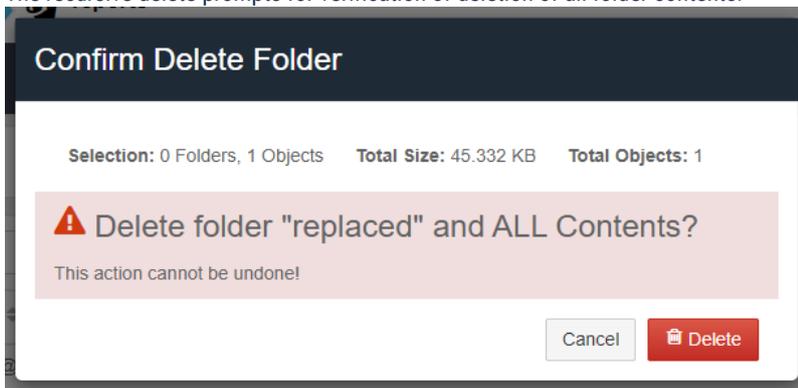
Recursive delete

The **Actions** menu provides the option to delete the folder while in the folder view. This delete is a recursive removal of all content, both objects and subfolders.

- The **Objects** count shows how many *content-bearing* objects are deleted along with the folder.



- The recursive delete prompts for verification of deletion of all folder contents.



Non-recursive delete

The ability to delete a persisted folder without deleting or renaming any content for a collection is present in the flattened view.

- Deleting the folder from the flattened view removes the folder and preserves the objects it contained, as well as the prefix in the name:

acme.ser... marketing... New Collection
admin1

Tip

The folder is temporary and appears dynamically because of prefixes on object names if the folder is not visible on a separate line in *this* view – there is nothing persisted in the cluster to be deleted.

Tip

The folder is temporary and appears dynamically because of prefixes on object names if the folder is not visible on a separate line in *this* view – there is nothing persisted in the cluster to be deleted.

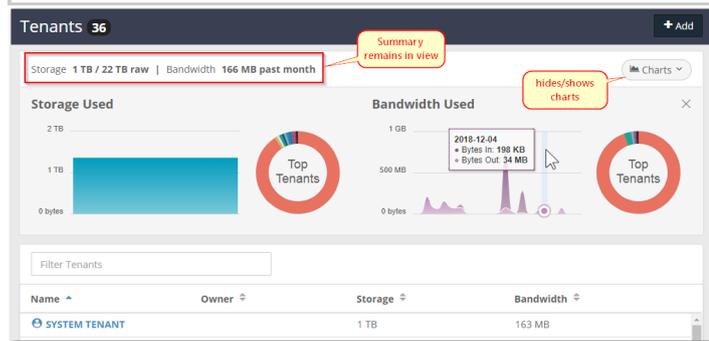
Usage Reports

- [Tip](#)
- [Storage Reports](#)
- [Bandwidth Reports](#)
 - [Note](#)

Each level of your Swarm site (tenants, domains, and buckets) includes a summary bar directly under the black title bar. This dynamic reporting and charting of bandwidth and storage usage is based on the data that is captured by [Content Metering](#). Historical usage queries populate these graphs, which make it easy for you to monitor usage status visually, from the dashboard.

Tip

Hover over any data point on a chart to see a pop-up of its details.



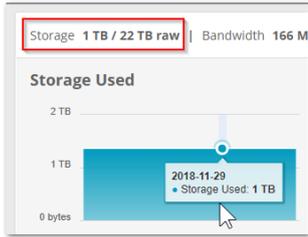
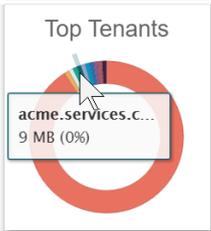
The summary bar alerts you to the Storage Used and Bandwidth Used status for the immediate context, so it is *always* visible. The **Charts** button expands and collapses the view of the bandwidth and storage usage charts that update dynamically for the tenant, domain, or bucket that is currently selected. Everything relates to the specific context being accessed; for example, if a user is allowed access to only a single bucket, she can see the storage and bandwidth status of her bucket, but *only* that bucket. Only those authorized to see the top-level **Tenants** view can see the report of the raw storage space that remains available in the cluster.

Errors - If the usage charts and totals are missing, it can have one of these causes:

- Metering is not enabled.
- Metering is enabled but without connection (which may be temporary).
- Metering data is not collected.

Storage Reports

The **Storage** reports show the amount of *capacity*. They are inclusive of all versions, replicas and erasure-coded segments that are stored at each level.

	
<p>Graph - A historical storage capacity report where the average of all storage data points on a given day is plotted over a rolling 30 day window.</p> <p>Used - The total amount of storage capacity in use for the given level.</p> <p>Root level only. The amount of storage capacity still available for use in the cluster.</p>	<p>Graph - The top 10 highest consumers of storage capacity in the given level at the time the chart was displayed.</p> <p>Percentage - The relative size of the displayed segment represents the percentage of the total for which a given consumer accounts. If one tenant uses half of the capacity in the cluster, that tenant is represented on the root level Top Tenants report as half of the circle.</p>

Bandwidth Reports

The **Bandwidth** reports show the amount of *activity*. They are inclusive of all bandwidth (both bytes in and bytes out) used at each level.

<p>Bandwidth 166 MB past month</p> <p>Bandwidth Used</p> <p>1 GB 500 MB 0 bytes</p> <p>2018-11-30 • Bytes In: 271 MB • Bytes Out: 11 MB</p>	<p>Top Tenants</p> <p>acme.services.C... 56 MB (2%)</p>
<p>Graph - A historical bandwidth report where the sum of all bandwidth data points on a given day is plotted over a rolling 30 day window. Bytes in and bytes out are represented by two different stacked colors in the chart.</p> <p>Used - The total amount of bandwidth used for the given level in the last 30 days.</p>	<p>Graph - The top 10 highest consumers of bandwidth in the given level over a rolling 30 day window. If there are more than 10 data points for a level, the 11th and all other data points are grouped into an Other segment.</p> <p>Percentage - The relative size of the displayed segment represents the percentage of the total for which a given consumer accounts. If one tenant uses half of the bandwidth total for the tenant over a 30 day period, that tenant is represented on the root level Top Tenants report as half of the circle.</p>



Note

Non-bucket content (tenanted unnamed objects) are associated with the bucket "Content IDs".

See [Content Metering](#).

Setting Storage Policies

- [Protection](#)
 - [Replication](#)
 - [Erasure Coding](#)
- [Versioning](#)
- [Policies for Unnamed Objects](#)
- [Override Alerts](#)

The following options are available to specify storage policies to apply to the objects it contains when editing the **Properties** of a domain or bucket:

Storage Policies

Inherit Protection
i

A domain or bucket inherits the protection settings in force above it (cluster, domain) by default.

i

Required permissions

Grant users these specific permissions: **PutDomain** and **CopyBucket** if users are allowed to change the content policies (replication, erasure coding, or versioning) on a domain or bucket through the Content UI.

- To see the options for a storage policy, deselect the **Inherit** checkbox, which expands the policy section.
- For guidance about the policy options, click the information icon, which toggles the help text.

Protection

Swarm allows flexibility in determining the type and level of content protection best fitting the storage needs. In Swarm storage, objects can be replicated and/or erasure-coded, with objects of both types co-existing in the same cluster.

Tip

Erasure coding helps cost-effectively scale clusters with many nodes and larger objects, while replication is better for smaller clusters and with smaller objects.

These settings allow a choice of replication and erasure-coding protection policies for the objects in this immediate context, subject to overrides by higher-level (cluster or domain) settings.

Replication

Replication protection requires the cluster to keep a specified number of copies (replicas) of each object.

- **Default Replicas:** Accept the inherited number or enter how many replicas are desired (subject to existing min and max values and query arguments).
- **Anchored:** Select to override any lower-level policies.

For more about replication in Swarm, see [Replication](#).

For implementation, see [Implementing Replication Policy](#).

Erasure Coding

Erasure coding (EC) protection divides very large objects in to multiple data (k) and parity (p) segments for distribution across k+p nodes, which is more space-efficient than storing very large replicas.

- **Enabled:** Select to allow erasure coding at this level and below (subject to higher-level policies).
- **EC Size Threshold (MB):** (not settable) Reports the object size that triggers erasure coding rather than replication.
- **Default Encoding:** Accept the inherited encoding, or enter data (k) and parity (p) values such as these examples:
 - 5 : 2 (1.4x footprint) - protection for 2 simultaneous disk failures.
 - 9 : 6 (1.7x footprint) - protection for 6 simultaneous disk failures and 1 subcluster failure in clusters of 3 or more subclusters.
- **Anchored:** Select to override any lower-level policies.

For more about erasure coding in Swarm, see [Erasure Coding EC](#).

For implementation, see [Implementing EC Encoding Policy](#).

Versioning

Swarm supports object-level versioning, which is a powerful content protection option that tracks, secures, and provides access to historical versions of objects, even after they are deleted. With versioning, applications can read, list, revert, and purge prior versions as well as restore objects deleted by mistake.



Best practices

- Plan for higher disk utilization with versioning: each update to a versioned object adds a new object to the cluster (one object updated twice results in three objects stored).
- Where possible, make use of lifepoints to control the lifetime – and thus the cost of storing – multiple versions of objects.
- For optimal resource management, limit versioning to the specific domains and/or buckets for which it is needed.

For more about versioning in Swarm, see [Object Versioning](#).

For implementation, see [Implementing Versioning](#).

The versioning state of the immediate context applies to every object in that context, without exception. Each domain and bucket has one of these versioning states:

disabled	(default) No versioning exists, so no versions are created. This state is the normal behavior of Swarm.	The Health Processor cleans up all residual prior versions, regaining space if changed from enabled to disabled in the domain or bucket. This feature (changing to disabled) is not available in Amazon S3.
suspended	No new versions are accumulated but old versions are retained. This is a hybrid between enabled and disabled that preserves history.	The chain of versions resumes from where it stopped if versioning is re-enabled from this state.
enabled	New versions are accumulated as they are created, starting with any version that exists at the time versioning becomes enabled for the object.	This is the only parent state from which domains and buckets can enforce a versioning policy that differs from its parent.
required	Domains only. Requires each of its buckets to have versioning enabled.	Use this to prevent bucket owners from disabling versioning policy.



Important

By default, versioning is disabled at the cluster and every other level. Versioning must be enabled for the cluster through the Swarm configuration setting `policy.versioning`. Cluster-level values are `disallowed` (default), `suspended`, and `allowed`. In a cluster with versioning allowed, every newly created domain and bucket starts with an unspecified state, so object versioning is disabled until enabled there *explicitly*.

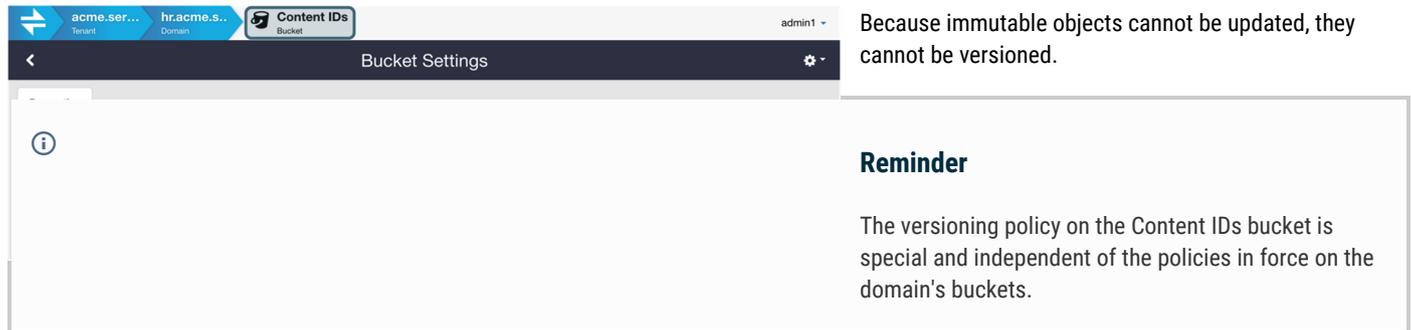
Reminder

The versioning option selected depends on what is permitted *in the given context*. Versioning occurs in a cluster that allows it but not in one that disallows it, such as a remote replication cluster if a domain has versioning enabled.

Policies for Unnamed Objects

All named objects are controlled by the policies on the buckets, but unnamed objects are handled separately for each domain.

To view and set the storage policy for unnamed objects in the domain, view **All Buckets** and open the special **Content IDs** bucket, which is the system-controlled container for all unnamed objects in the domain. Open the settings (cog icon) and specify the policy to apply to the unnamed objects in the domain:



acme.ser... hr.acme.s... Content IDs Bucket admin1

Bucket Settings

Because immutable objects cannot be updated, they cannot be versioned.

Reminder

The versioning policy on the Content IDs bucket is special and independent of the policies in force on the domain's buckets.

Override Alerts

Because the defined policies are subject to override by policies at higher levels (such as the cluster settings), alert icons and messages inform if the specified policy is blocked from going in to effect.

i

Important

Policy settings are still *valid*, even if they do not go in to effect immediately. A policy request may be implemented if the higher-level policy changes at a future time or if content is replicated to a cluster with different policies.

Storage Policies

Inherit Protection ⓘ

REPLICATION

Default Replicas (2.0x footprint)

ERASURE CODING

Enabled

EC Size Threshold (MB)

EC Default Encoding : (1.5x footprint) ▲

Video Clipping for Partial File Restore

- [Installing Video Clipping](#)
- [Creating a Clip](#)
- [Metadata for Clips](#)
- [Monitoring Clipping](#)
- [Audit Logs for Clipping](#)

Once you upload a video into a bucket in Swarm, you can view and share it from the Content UI. Then, if you have the optional **Video Clipping** tool installed, you can also excerpt out portions and store them as new, standalone videos within Swarm. This is part of the functionality commonly referred to as *partial file restore* in the media and entertainment industry. (v6.3)

These are types of activities where video clipping can prove critically helpful:

- **Sporting Event Analysis**
A sporting team uploads its hours-long recording of a recent game into Swarm. It wants key highlights (typically 2 minutes or less) to go to broadcasters, commercial sponsors, or coaches for further consumption. Rather than send the entire (very large) video file with timestamps, the owner of the video can use the Content UI to extract the relevant segments into small video clips, which are then sent out as needed.
- **Targeted Post-Production**
A media company needs to have special post-production work applied to a given scene or set of takes, but it wants to avoid transmitting the full set of video (multiple GB) to the post-production shop. Instead, the required portion of video, buffered with a few seconds of video before and after, is extracted into a smaller (measured in MB) clip and sent for processing.
- **Presentation Highlights**
A professional organization uploads its full hour-long conference presentations into Swarm, then creates highlight excerpts to use on the website, in social media, and in marketing materials.

Installing Video Clipping

After you obtain the optional package for video clipping, you can install it on your Gateways; no configuration is required. In addition to the tool RPM, you will install the supporting framework for multimedia formats, FFmpeg, which Swarm provides preconfigured within a Docker container.

Best practice

Use Docker and install the provided container; this spares you dealing with FFmpeg repo choices, its numerous dependencies, and additional resource changes required to protect Gateway's performance. Installing [FFmpeg](#) directly requires additional configuration that you would need to complete with DataCore Support.

1. Copy the package for video clipping to a server that is running Content Gateway:
`caringo-videoclipping-VERSION.noarch.rpm`
2. If the server is running RHEL/CentOS 6.x, upgrade to RHEL/CentOS 7 (required by Docker).
3. If it is running Content Gateway 6.0 or earlier, upgrade Gateway (versions 6.1 and higher support drop-in features such as video clipping).
See [Gateway Installation](#).
4. If Content UI is version 6.1 or earlier, upgrade it to support this feature.
See [Content UI Installation](#).
5. Install the package.

```
yum install caringo-videoclipping-VERSION.noarch.rpm
```

The installation creates a `features.d` directory under `/etc/caringo/cloudgateway/`, which is where Gateway detects optional, dynamic features.

6. Install multimedia support via FFmpeg, which Swarm provides preconfigured in a Docker container.

From an Internet-connected machine:

- a. Check whether Docker is installed: `docker info`

If not, install the Docker package, then start the daemon, check its status, and enable it system-wide. See docs.docker.com/install/linux/docker-ce/centos/

- b. Verify Docker by running a container test:

```
docker run hello-world
```

- c. Load the provided container for FFmpeg:

```
docker load < caringo-gateway-VERSION.feature.ffmpeg.via.docker.VERSION.x86_64
```

7. Gateway will create the following directory for spooling video clips: `/var/spool/caringo/cloudgateway/features`

If it cannot create this directory, Gateway will refuse to start.

8. In setting `iptables` rules, Docker closes external TCP access to port 80 (although ping and ssh still work); be sure to explicitly open port 80 again so that clients can access Gateway.

```
firewall-cmd --add-port=80/tcp --permanent
firewall-cmd --reload
systemctl restart docker
```

9. Repeat the above process on any remaining Content Gateway servers.

10. Restart the Gateway(s).

On reboot, Content Gateway detects the feature; on a page refresh, Content UI displays the clipping control for video content.

Creating a Clip

Notes

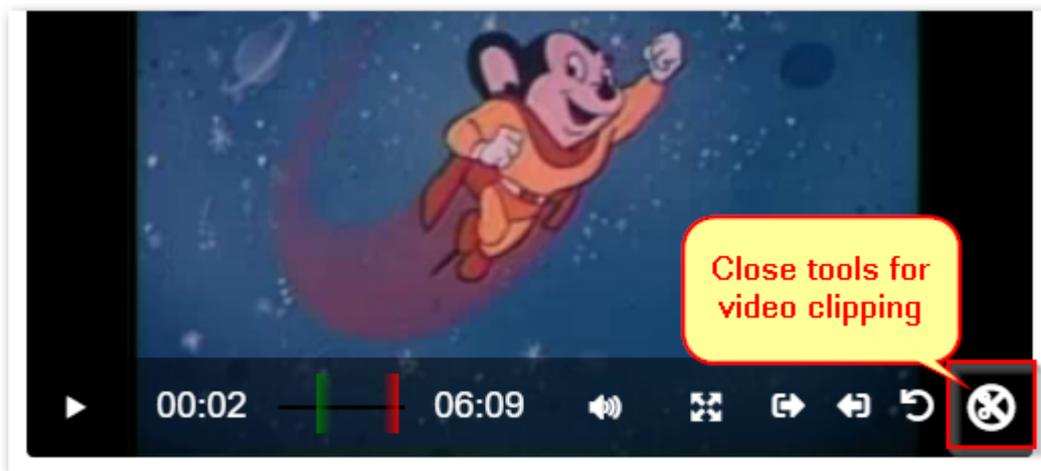
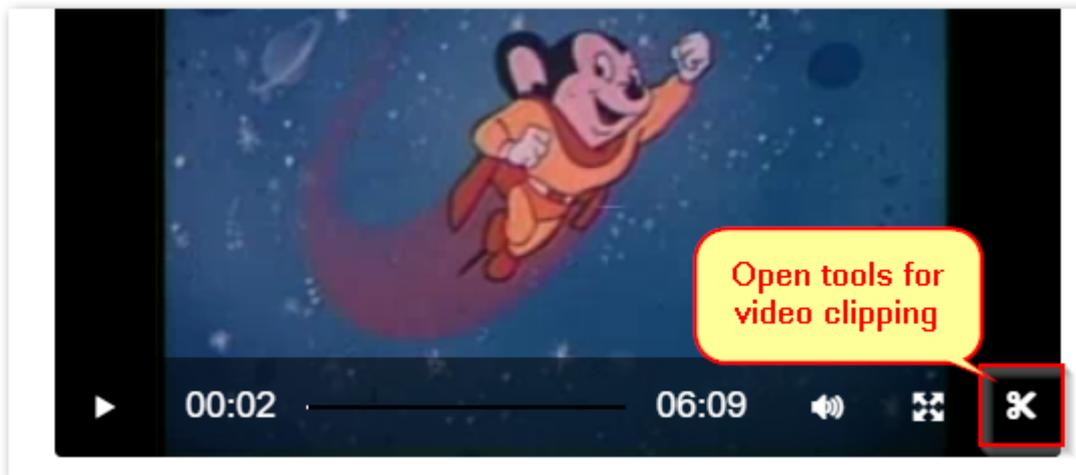
Currently, only named objects can be clipped, and only one clip can be saved at a time.

The browser-based tool requires an [HTML5-supported video](#) format; MPEG-4/MP4 (H.264) and WebM have been tested and are supported. Video clips are output as MP4 format.

If you have Video Clipping installed on Content Gateway, the Content UI automatically detects the installed feature. Whenever you view video-format objects with sufficient frames to be clipped, the Content UI adds the scissor



icon to the right of the video control bar. This is a toggle command: click it once to open the video clipping toolbar, and click it again to hide the clipping commands:



Tip

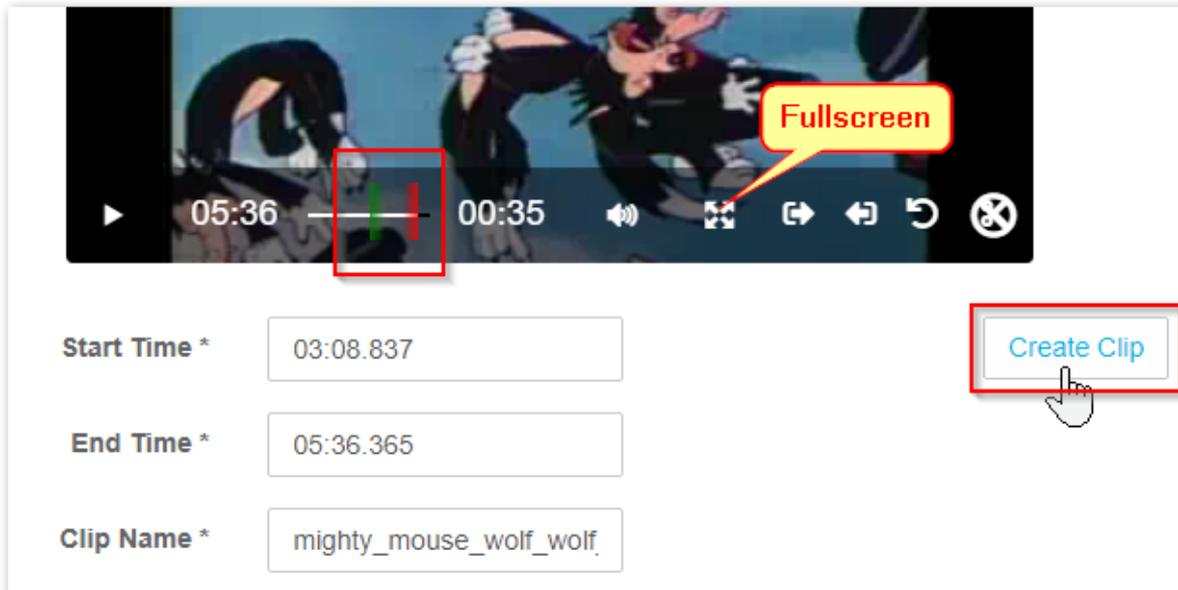
For higher resolution, use **Full Screen** mode:



You have three ways to set the time span of the clip, depending on the precision you need:

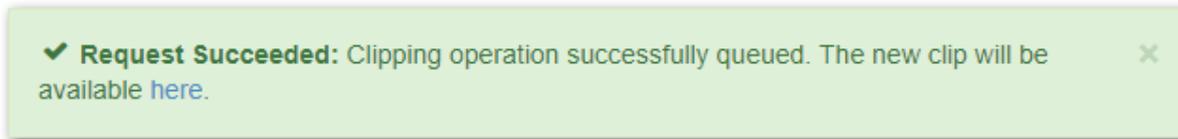
- Drag the **green** and **red** bars
- Select a bar and use the left and right arrow keys to move it frame by frame (supported in Chrome)
- Enter explicit *minute:second[millisecond]* values in the **Start Time** or **End Time** fields

As you adjust the start and end times, the tool dynamically appends those values to the source filename to create the **Clip Name** (unless you have put a custom value in the field). This becomes the object name when you click **Create Clip**:



The screenshot shows a video player with a progress bar. A red box highlights the start and end time markers on the progress bar. A yellow callout bubble labeled 'Fullscreen' points to the fullscreen icon. Below the player is a form with three fields: 'Start Time *' (03:08.837), 'End Time *' (05:36.365), and 'Clip Name *' (mighty_mouse_wolf_wolf). A 'Create Clip' button is highlighted with a red box and a hand cursor.

The tool saves it into the *same bucket* as the source video:



A green banner with a checkmark icon and the text: "Request Succeeded: Clipping operation successfully queued. The new clip will be available [here](#)." A close button (X) is in the top right corner.

The default naming creates a file that shows `{original-name}-{start}-{end}.{filetype}`, but you can override the **Clip Name** field entry to follow any naming conventions that your organization needs:

Name ▲	Owner ⇅	Storage Date ⇅	Size ⇅
<code>{name}-{start}-{end}</code> film-16m01s-43m35s.mp4	admin1	Standalone video	2.14 MB
film.mp4	admin1@	2019-07-26 4:24:39 PM	575.27 MB

Tips

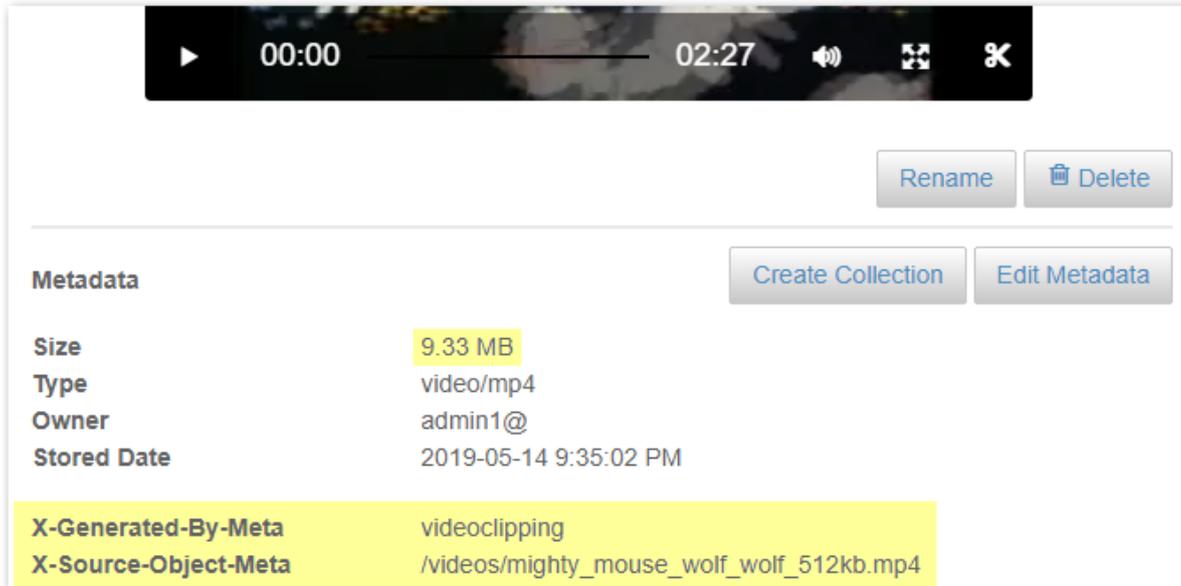
File format – The fastest clip generation is from source videos that are in MP4 format, which matches the clip format that is output by Content UI. Clipping from WebM-formatted source videos requires additional transcoding.

Delays in listing – Even though smaller clips generate quickly, the process is asynchronous, and the bucket listing will update as soon as possible. For large clips that take time to process, you might first see a JSON file appear in the bucket listing ahead of the clip; this file contains information about the clip you created, and its appearance indicates that the write succeeded. These JSON artifacts have brief lifespans to enable auto-cleanup, but you can edit the metadata and delete the lifepoint if you want to retain it. See *Audit Logs for Clipping*, below.

As soon as you view your new clip, you can use the **Share** menu to distribute it to recipients:

Metadata for Clips

When you open your new video clip, you see that the **Size** reflects the video range you selected; this clip is a standalone video, not a stub pointing to a range in the original (and therefore no dependency on it):



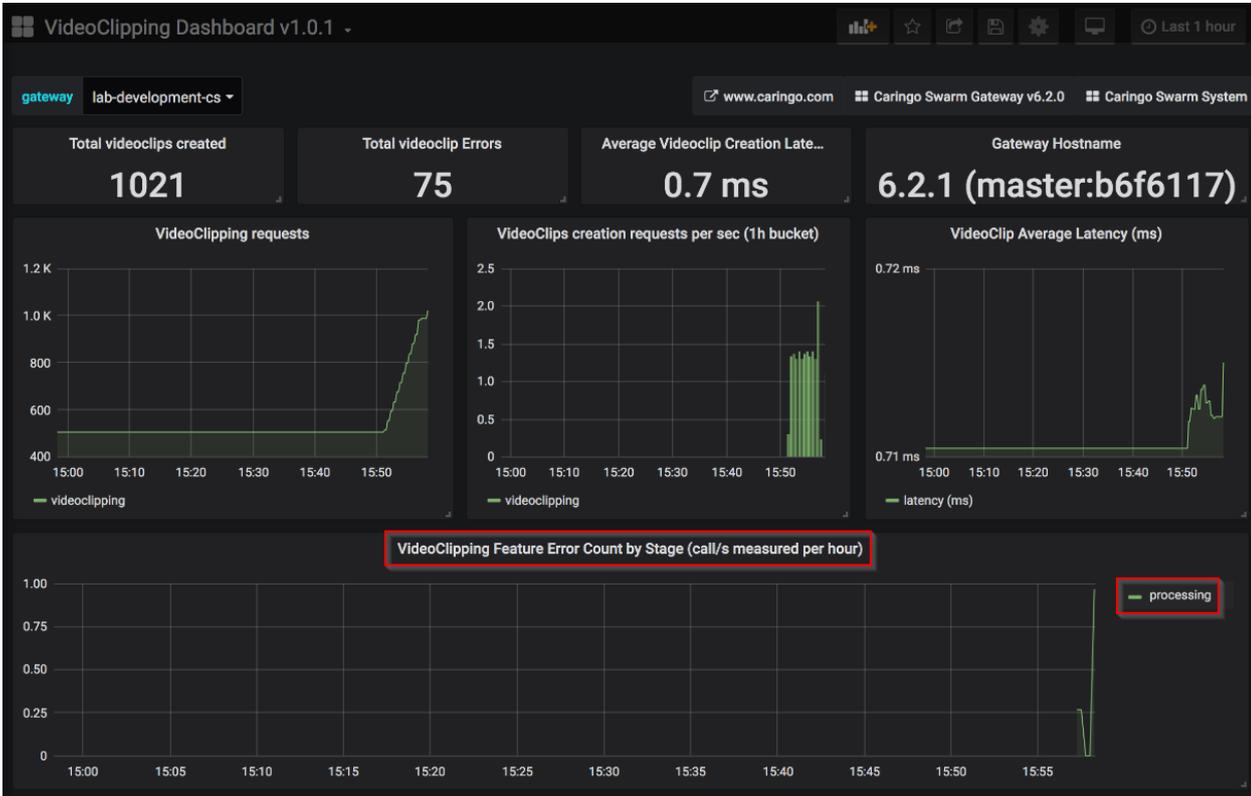
Metadata	
Size	9.33 MB
Type	video/mp4
Owner	admin1@
Stored Date	2019-05-14 9:35:02 PM
X-Generated-By-Meta	videoclipping
X-Source-Object-Meta	/videos/mighty_mouse_wolf_wolf_512kb.mp4

The object is tagged with the following metadata when it is written:

- **x-owner-meta** – The user that initiated the clipping request, *not the owner of the source video*.
- **x-generated-by-meta** – The name of the feature that generated the object.
- **x-source-object-meta** – The SCSP URL (stripped of host and protocol) for the original source object, *regardless of whether it still exists as such in Swarm*.

Monitoring Clipping

To monitor for problems, you can implement Swarm's [Prometheus Node Exporter and Grafana](#) and install the latest **Caringo VideoClipping Dashboard** (<https://grafana.com/grafana/dashboards?search=caringo>):



Clipping has three distinct stages:

1. **Preprocessing** – Before any processing starts, Gateway checks to ensure that the request is valid. It verifies conditions such as these:
 - The clipping addon is installed
 - The required parameters are set
 - The current user has access rights
2. **Processing** – Here Gateway runs the request. This involves extracting the needed portion of the input and creating the clip on the local disk.
3. **Postprocessing** – At this point, Gateway uploads the clip as a new video object in Swarm. This new video references but has no dependencies on the original.

By noting which of these three stages your errors occurred in, you can narrow down the scope of the cause. For details, check the audit logs.

Tip
Preprocessing errors are the most common, and they most likely result from configuration/authentication errors that can be quickly resolved.

Audit Logs for Clipping

Each video clipping event logs multiple operations to provide auditing through the process, which might take a while to complete. When you create a video clip, Gateway handles it asynchronously (to accommodate requests of all sizes) and acknowledges the request with an INVOKE message, which will appear *first* in your audit log. That acknowledgement references the future JSON result object. When that JSON result later posts, it reports the outcome of the clipping request, such as the ffmpeg exit code and the duration, capturing the same information that you would get back on the response if it were synchronous.

```
2019-08-22 14:32:04,991 INFO [F38143E84D3EC62E] 2 192.168.1.154 192.168.1.154 Feature:videoclippi
2019-08-22 14:32:15,022 INFO [F38143E84D3EC62E] 2 192.168.1.154 192.168.1.154 Feature:videoclippi
2019-08-22 14:32:15,061 INFO [F38143E84D3EC62E] 2 192.168.1.154 192.168.1.154 Feature:videoclippi
```

All JSON result objects are temporary, by default: they are created with a lifepoint that triggers deletion after 5 days. You can change the default in the [Gateway Configuration](#), `gateway.cfg([dynamic_features] responseObjectLifetime=5)`.

See [Gateway Audit Logging](#).

Configuring Tenants

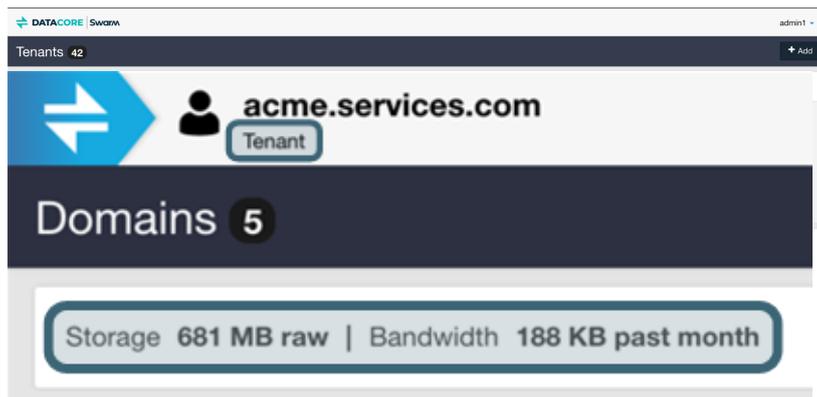
- [Tenant Essentials](#)
- [Tenant Provisioning Steps](#)
- [Tenant Properties](#)
- [Permissions](#)
- [Tokens](#)

Tenant Essentials

The concept of the **tenant** relates only to Gateway, not to Swarm Storage. Within a cluster, a tenant is the primary entity for dividing and controlling both access and resources. These are the critical features:

- **Ownership.** Each tenant owns one or more Swarm storage domains.
- **Access control.** Tenants can define separate identity management systems so the users and groups within them are separated from those in other tenants.
- **Delegation.** Tenant administrators can create and access storage domains on behalf of the tenant and they can delegate management duties for the storage domains they create.
- **No content.** The tenant does not itself store end-user data; it is only a container for meta information about the tenant, users, and storage domains.

Tenant Usage – The **Storage Used** chart displays the current size of the storage footprint used by all tenants, inclusive of all versions, replicas, and erasure-coded segments when viewing *all* tenants in a Swarm instance. The **Bandwidth Used** chart displays the total bandwidth (both bytes in and bytes out) used by each tenant over a rolling 30-day window. See [Usage Reports](#).



When opening up a tenant, a reports the usage at the very top, along with the total domain count:

Dynamic Filtering – All columns are sortable either ascending or descending with a default sort on the tenant name. Narrow the listing by entering a string in the **Filter** box, which filters by **Name** if a large number of tenants exist:

<p><i>(Information icon)</i></p>	<p>SYSTEM TENANT</p> <p>Note the default system-managed SYSTEM TENANT always displays at the start or end of the list and not in alphabetical order.</p>
<p><i>(Information icon)</i></p> <p><i>Warning:</i> This command cannot be undone, so proceed with caution.</p>	<p>Delete Tenant</p> <p>The Delete command deletes the tenant and <i>all</i> domains, including the buckets and uploaded contents.</p>



Tenant Provisioning Steps

These are the typical steps when provisioning a tenant. Details for performing these steps are documented later in this guide.

1. Create the tenant.
2. Optionally,
 - a. Assign ownership of the tenant.
 - b. Configure the tenant's identity management system.
 - c. Configure the tenant's access control policy.
 - d. Configure the tenant's quota.
3. Create one storage domain within the tenant to be used by the tenant's owner or primary user.
4. Assign ownership of the storage domain to the tenant's owner or primary user.
5. Provide a login URL for the storage domain to the tenant's owner or primary user.

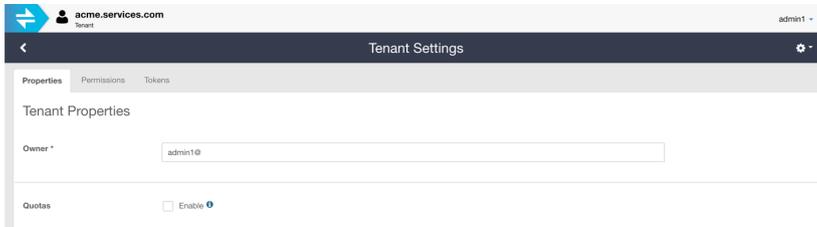
The format of the login URL is described in [Using the Content UI](#).



Important

The user must have a starting storage domain to log in to the Content UI when delegating tenant authority to a user.

Tenant Properties



Owner

Every tenant must have an owner, who has access to and ultimate authority over it. As a root admin, create a tenant for another to manage, as a tenant admin can create a domain for another to manage.

Note

Ownership defaults to the specific root administrator who created the tenant, but the owner does not need to be a root administrator.

Change the owner when creating a context for someone else to manage. One does not want to own or be responsible for managing the data in the tenant when creating a tenant for a client.

Quotas

Quotas can be set to determine how much storage and/or network bandwidth the tenant is permitted to consume.

See [Setting Quotas](#).

Identity Management

The IDSYS objects define the identity management systems controlling the tenant's users:

- User and group information
- The authentication system

See [Setting Identity Management](#) and [Gateway Identity System](#).

Permissions

Permissions are determined by the access control policy, which are the rules granting (or denying) users and groups the ability to perform specific actions.

See [Setting Permissions](#) and [Gateway Access Control Policies](#).

Tokens

See [Setting Tokens](#).

Metadata Encoding

Any non-ASCII characters in metadata are encoded for storage. Swarm encodes header field content according to the rules of [RFC-2047](#) to comply with HTTP/1.1 per [RFC-2616](#). Characters in sets *other* than ISO-8859-1 must be encoded.

Encoding of non-ASCII characters may appear in the metadata details for an object viewed in the Content UI:

Original filename: Bébé Cerf.jpg

Rename Delete

Create Collection Edit Metadata

Size	2.30 MB
Type	image/jpeg
Owner	admin1@
Stored Date	2021-04-02 7:45:57 PM
X-Last-Modified-By-Meta	admin1@
Castor-System-Cid	44b2dbbabd89
Castor-System-Cluster	jades
Castor-System-Created	2021-04-02 7:45:57 PM
Castor-System-Name	B%C3%A9b%C3%A9%20Cerf.jpg
Castor-System-Version	1617392757.208
Content-Disposition	inline; filename*=UTF-8' 'Be%CC%81be%CC%81%20Cerf.jpg

Non-ASCII characters in metadata are encoded for storage

Swarm allows all string-typed header fields to have multiple lines as well as encoded words. Swarm stores the header value as-is with the object metadata. It decodes the value when Swarm needs to use that value (such as for metadata indexing). Swarm decodes header fields into Unicode and then operates on the decoded values. The original encoded persistent headers remain safely stored with the object and are returned when the object is retrieved.

See [Encoding Non-ASCII Characters in Metadata](#) for Swarm's handling of metadata.

Setting Remote Synchronous Write (RSW)

- [What is Synchronous Write to Remote Sites?](#)
 - [RSW](#)
- [Implementing Remote Targets](#)
- [Enabling Remote Synchronous Writes](#)
 - [Domains](#)
 - [Buckets](#)
- [Managing RSW Manually](#)
- [Logging for Remote Writes](#)

What is Synchronous Write to Remote Sites?

Swarm's [replication feeds](#) are mature and resilient, with each feed syncing content iteratively and asynchronously, linking one source cluster to one target destination. Replication feeds keep target clusters continuously updated with object changes in the source cluster, and having multiple replication feeds allows distribution of content across a set of remote Swarm clusters.

Starting with Swarm 12, configure individual domains and buckets to perform an object write across all remote sites at the same time. Write completion to a given bucket or domain is delayed until *every* remote replication is completed when remote synchronous writes are enabled. There are two situations which need this synchronous write to remote sites:

- An application requires assurance that backups (replicas) are committed to every site.
- Content publication needs require newly added content be readable from *any* remote site immediately upon write completion.

Remote Site Synchronous Write – or *Remote Synchronous Write (RSW)* – is the option to have a client create/update a Swarm object and have that change committed in *all active remote* clusters immediately, before the client receives a success code. An object written this way has the same UUID and metadata across all Swarm clusters as if it is replicated normally. It makes use of the [SCSP SEND](#) method. (v12.0)



RSW

Synchronous Write to Remote Sites is the same as **Remote Synchronous Write (RSW)**. All commands and log messages use the acronym **RSW**.

This is separate from [Replicate on Write \(ROW\)](#), which guarantees immediate replication within the *same* cluster.

Implementing Remote Targets

Remote Synchronous Write engages as many replication targets as are active *at the time of the write*. Start using remote synchronous write with one remote cluster and add additional remote clusters later. Remove or change remote sites in the future while continuing to use RSW.

1. Complete the upgrade to Swarm 12 or higher, including the latest versions of Content Gateway, Content UI, and Swarm Storage. Current versions are required for this feature, but no additional components need to be installed.
2. Complete provisioning and initialization for the additional remote Swarm clusters participating in remote synchronous writing.
 - [Content protection policies](#) do *not* need to match across all clusters. Versioning may not be enabled (which enlarges the footprint) on any but the primary cluster.
3. Create separate replication feeds to each of the remote Swarm clusters in the primary (source) cluster. See [Replication Feeds](#) for details.

- Navigate to **Cluster > Feeds**, select **+ Add**, and select **Replication** in the Swarm UI.
- Leave the legacy setting **Propagate Deletes** enabled.
- Choose **Replicate via direct POST**, rather than the legacy GET mode for the remote cluster's **Replication Mode**.

4. Note: replication-type feeds with a Status of **Active** participates in RSW, if enabled:

Feed	Status	Events queue	Deletes queue	Rate	Scope	Target
Replication rep-feed	Active	18,622,266	881	460.7 /hour	Domain	jans222 (POST)
Search – Primary navys	Active	0	0	1,460.7 /hour	Global	navys

- Set up replication feeds in the remote clusters if the remote clusters are accepting content to be mirrored back to the primary cluster.
- Verify the feeds are running successfully in each of the target clusters, populating objects from the source cluster.

Enabling Remote Synchronous Writes

Click the **Settings** (gear icon) and set the **Properties** to have in force for that domain or bucket in the Content UI. The policy change takes effect on **Save**.

Constraints – As with the Storage Policies (replicas, encoding, versioning) for domains and buckets, this setting is constrained by the cluster's configuration. The constraints are whether the cluster has any replication feeds that are configured and whether they are in **Active** status for Remote Sites; no synchronous writes occur if there are not any remote clusters being replicated to.

Domains

The option *must* be enabled at this level if synchronous writes to remote sites to include **unnamed objects** (which do not reside in buckets) is desired.

Best practice is to enable the option at the level of those buckets if synchronous writes for certain buckets is desired.

- **Synchronous Write Enabled** – Toggle to explicitly enable or discontinue the feature.
 - For a domain, enabling it means that *each of its buckets* also default to having it enabled, unless inheritance is overridden.

Bucket Properties

Owner *

Quotas Enable ⓘ

Storage Policies

Inherit Protection ⓘ

Inherit Versioning ⓘ

Remote Sites Inherit Policy ⓘ

SYNCHRONOUS WRITE

Enabled

Buckets

Disable **Inherit Policy** and explicitly enable it for *each* affected bucket unless enabling synchronous write at the domain level:

- **Inherit Policy** – (*default*) Select to accept the domain's policy, which is disabled by default until the domain *explicitly* enables it.
 - With inheritance off, the bucket does not respond to domain-level changes.
- **Synchronous Write - Enabled** – Toggle to explicitly enable or disable the feature.

Managing RSW Manually

Manage and verify remote synchronous write settings from the command line:

- To **enable** remote synchronous write, run a curl command that specifies *one domain or bucket* that is included in remote synchronous writes:

```
curl -X PUT http://<Gateway>/_admin/manage/tenants/_system/domains/<domain>/rsw?state=enable
curl -X PUT http://<Gateway>/_admin/manage/tenants/_system/domains/<domain>/buckets/<bucket>
```

Repeat the RSW enabling command on each remaining domain and/or bucket that is part of remote synchronous write.

- To **disable** remote synchronous write from the command line, run a curl command that specifies *one domain or bucket* that is excluded in remote synchronous writes:

```
curl -X PUT http://<Gateway>/_admin/manage/tenants/_system/domains/<domain>/rsw?state=disab
curl -X PUT http://<Gateway>/_admin/manage/tenants/_system/domains/<domain>/buckets/<bucket>
```

- To **inherit** the domain setting again after having had it disabled on the bucket, change the state back to `unset`:

```
curl -X PUT http://<Gateway>/_admin/manage/tenants/_system/domains/<domain>/buckets/<bucket>
```

- To **verify** the RSW status of a given domain or bucket, run the RSW command:

```
curl -I http://<Gateway>/_admin/manage/tenants/_system/domains/<domain>/rsw
curl -I http://<Gateway>/_admin/manage/tenants/_system/domains/<domain>/buckets/<bucket>/rsw
```

Logging for Remote Writes

A file is completely written to the original cluster and then sent to the remote clusters before the client is notified of completion, so this incurs some latency for the request when a large file is written with remote synchronization. The new object is accessible from the source cluster even if errors prevented immediate replication in any remote clusters.

In Gateway log messages, find Remote Synchronous Write statuses by looking for the prefix "RSW: ".

At the **INFO** level of logging, a remote replication report displays:

```
2020-06-17 23:59:03,396 INFO [qtp412788346-59|9403B57E570AFA2C] RSW: domain1/bucket1/object1 repl
```

When enabling **DEBUG** levels (see [Gateway Logging](#)), view all detail statuses, which show that the remote writes are resolved before the RSW completes:

```
2020-06-17 23:59:03,396 DEBUG [qtp412788346-59|9403B57E570AFA2C] RSW: Received RSW final response
2020-06-17 23:59:03,396 DEBUG [qtp412788346-59|9403B57E570AFA2C] RSW: -- header Feed-1-Status = 0
2020-06-17 23:59:03,396 DEBUG [qtp412788346-59|9403B57E570AFA2C] RSW: -- header Feed-1-StatusTime
2020-06-17 23:59:03,396 DEBUG [qtp412788346-59|9403B57E570AFA2C] RSW: -- header Feed-2-Status = 0
2020-06-17 23:59:03,396 DEBUG [qtp412788346-59|9403B57E570AFA2C] RSW: -- header Feed-2-StatusTime
2020-06-17 23:59:03,396 INFO [qtp412788346-59|9403B57E570AFA2C] RSW: domain1/bucket1/object1 repl
2020-06-17 23:59:03,396 DEBUG [qtp412788346-59|9403B57E570AFA2C] ScspProxyRequestHandler: RSW Res
```

The feed numbers refer to the **ID** field of the feed definition. `Feed-1-Status = <number>` is a count; how many objects are *remaining* for the feed to handle.

- 0 means the feed is empty (completed).
- 1 means a timeout occurred.
- 2 or higher indicates an error.

The **RSW Result** is 1/2 or 0/2 instead of 2/2 if anything remains in the feed that cannot be remotely written.

Editing Names, Metadata, and Versions

- [Renaming Objects](#)
- [Editing Metadata](#)
- [Viewing and Deleting Versions](#)

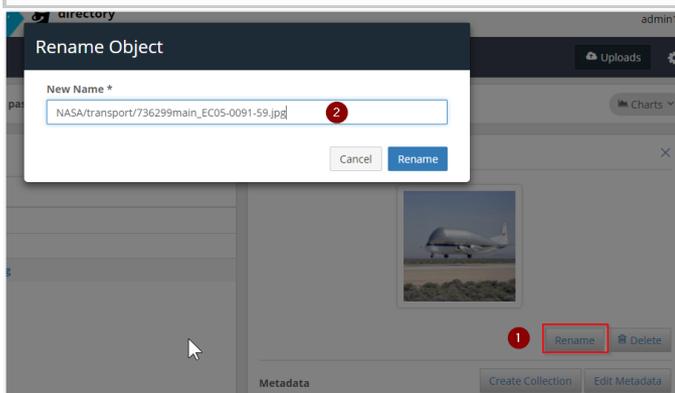
Renaming Objects

When selecting and viewing an individual object, change the object name using the **Rename** command. (v5.5)

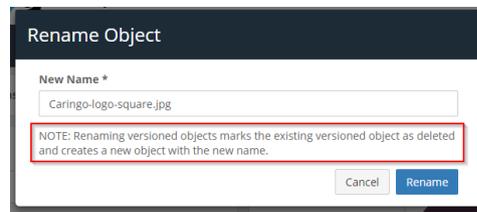
The name includes any pseudo directories added on upload to Swarm, as well as file extensions retained.

Tip

The original filename of the uploaded object is always preserved in the metadata. Under the **Metadata** section, select **more** and locate the **Castor-System-Name** field.



Versioned objects – Renaming a versioned object marks the existing object as deleted (which ends one chain of historical versions) and creates a new object using the new name (which begins a new chain of historical versions). A reminder of the impact of versioning displays when attempting to rename a versioned object:



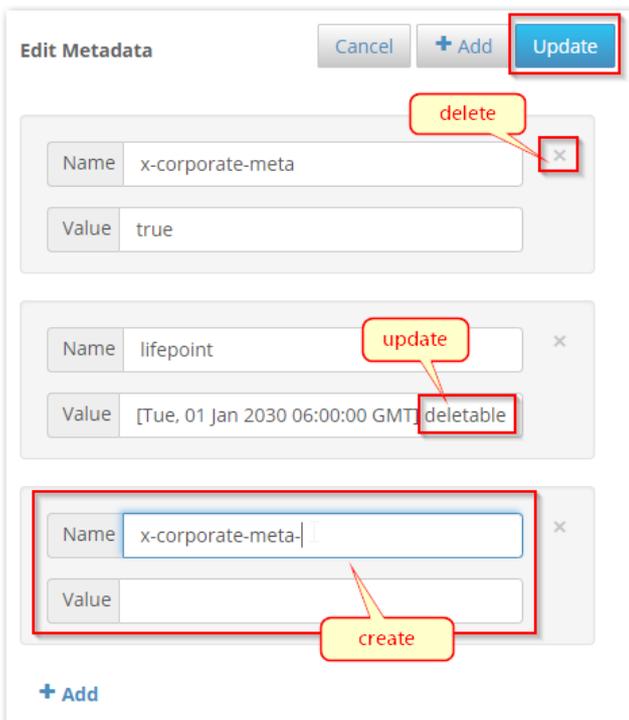
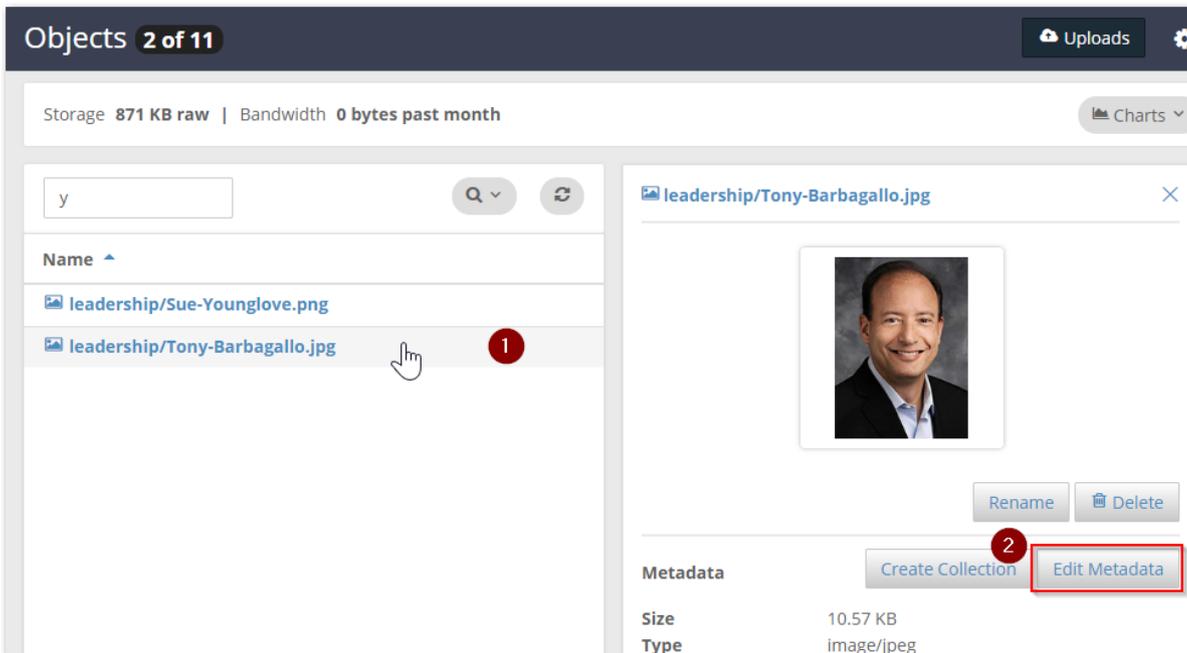
Editing Metadata

The Content UI allows addition and editing of custom object metadata

directly in the UI, unless the object is immutable (see [Understanding Unnamed Objects](#)). While applications can also add and change object metadata, this interface allows correction and extension of metadata for a given object from the convenience of a browser. If an editable metadata field (such as `x-acme-meta-color`) exists for the object, it appears in the object's detail view, and the value (`red` to `blue`) can be changed or removed entirely. (v5.4.0)

Versioning

Metadata for the *current* version of the object is edited if versioning is enabled for the object's domain or bucket. One new version is created if several changes are made to the metadata and saved. Another version is created, with both sets of metadata if another metadata change is made and saved.



What metadata is editable?

The Content UI only exposes metadata that has been created by the organization or is otherwise appropriate to change.

- *Fields that are system-controlled or generated by the request are not editable.* The ETag (entity tag) is a 128-bit number used to uniquely identify the object on the Internet, and it must not be altered by hand.

- *Custom metadata names are protected from overwriting.* Update the **Value** of an existing header, but if the **Name** is updated, a new header is appended to the object and the original entry is preserved. To replace an existing header, create a new one with the correct name and value and delete the old one.

These are fields that are available for adding, editing and deleting:

	Examples	Exceptions	Notes
Content-*	Content-Language Content-Location Content-Type	Content-Length Content-MD5	Validate changes when updating required metadata, such as Content-Type. Swarm restores the last known value if attempting to delete required metadata.
Castor-*	Castor-OLD-Metadata	Castor-System-*	These custom fields use a deprecated format, but they may still appear on older objects.
X-*-Meta	X-Country-Meta X-Zip-Postal-Code-Meta X-lat-long-Meta		
X-*-Meta-*	X-Zer0-Meta-code X-Zer0-Meta-name X-Zer0-Meta-division		

Creating metadata

The Content UI verifies the field name is valid for custom metadata in Swarm when selecting the **Add (+)** command to create new metadata. Follow these rules to name metadata:

- Use letters with any mix of case.
- Only use numbers and dashes (-) *following* letters.
- Create unique field names (no duplicates allowed)

The case entered is preserved; `x-foo-meta` and `X-Foo-Meta` are the same field.

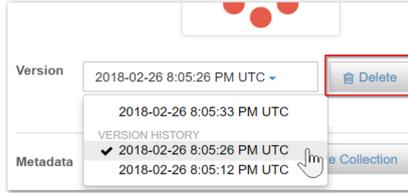
Viewing and Deleting Versions

Although metadata cannot be edited on historical versions (because they must not be altered in any way), the ability to delete them is available. The Content UI supports precise deletions of one or all historical versions of a given versioned object. (v5.4.0)

Viewing – Click on the drop-down arrow in the Version list to view and access the object's version history. The preview image above an object version updates when a prior version of an object is selected. Click on the thumbnail to open the historical version in another tab. Download it from there if needed.

Deleting – A prompt appears to verify which kind of delete to perform when **Delete** is selected for a versioned object:

- **Delete the object**, which writes a delete marker as the new current version. Deleted objects can be restored, if needed.
- **Delete the current version**, which removes the selected version from the version history. The deleted version cannot be restored.
- **Delete all versions**, which removes the current version and the entire version history. These deletions cannot be restored.



The screenshot shows a user interface for version management. At the top, there are three red circles. Below them, a 'Version' dropdown menu is open, showing a list of timestamps: '2018-02-26 8:05:26 PM UTC', '2018-02-26 8:05:33 PM UTC', and 'VERSION HISTORY'. The first item is selected. To the right of the dropdown is a 'Delete' button. Below the dropdown is a 'Metadata' section with a 'Collection' button. A mouse cursor is hovering over the 'Collection' button. The 'Metadata' section also contains a list of timestamps: '2018-02-26 8:05:26 PM UTC' (checked) and '2018-02-26 8:05:12 PM UTC'.

Using the Content UI

- [Tip](#)
- [Important](#)
- [Accessing the Content UI](#)
 - [Note](#)
 - [Tip](#)
- [Creating a tenant](#)
 - [Note](#)
- [Creating a storage domain](#)
- [Creating a bucket](#)
- [Uploading content](#)
- [Creating a search collection](#)
- [Resources](#)

The Content UI allows creation, viewing, and management of tenants, domains, buckets, and objects from the convenience of a web browser.

 **Tip**

For the best experience using the Content UI, use the latest [Chrome](#) or [Firefox](#) browsers.

The breadcrumb navigation at top shows the location in the hierarchy (which is how access to content is segregated and controlled). The hierarchy is based on one-to-many relationships:



 **Important**

What is visible in the Content UI is controlled and protected by access permissions. A Domain admin is not able to see anything (domains, tenants, clusters) outside of the domain they are authorized to manage.



The DataCore icon link attempts to take users to the top-most level (domain, tenant, or root) allowed for them by *applied* permissions. It takes them to the storage domain they logged in to (the URL in the address bar); the user must navigate to the root level, even if the permissions do not allow them to list if the hostname is not a storage domain.

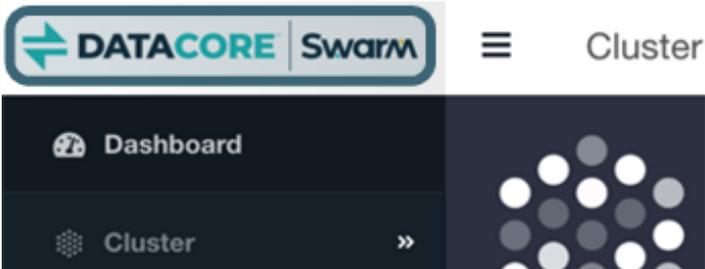
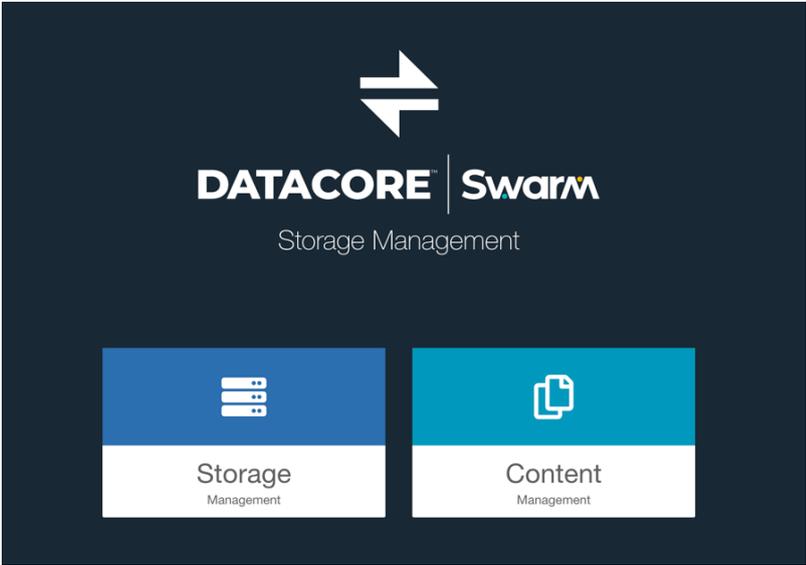
See [Content Gateway Concepts](#).

Accessing the Content UI

The admin users for the Content UI are defined during the [Gateway installation](#), via a root policy configuration file that grants full cluster permissions to a specific LDAP group or list of users.

**Note**

An authorized user must login. There is no anonymous access to the Content UI.

<p>Cluster</p>	<p>Browse to the cluster's Content UI and use the login name and password created for the Root administrator to log in as a cluster administrator:</p> <pre>http://GATEWAY·IP:SCSP·BINDPORT/_admin/portal http://GATEWAY·IP:80/_admin/portal (default)</pre> <p>The <code>SCSP·BINDPORT</code> refer to <code>bindPort</code> settings in the Gateway Configuration, in the <code>[scsp]</code> section.</p> <div data-bbox="268 432 1485 1556" style="border: 1px solid #ccc; padding: 10px;"> <p>Tip</p> <p>Navigate to the Content UI directly by clicking the top left Swarm logo to reach the sitemap if logged in to the Swarm UI:</p>   <p>The Content UI opens to the correct port, as defined in the Gateway Configuration when Content is clicked.</p> </div>
<p>Tenant or Domain</p>	<p>To log in as a tenant or domain administrator, browse to a domain's Content UI and log in with the admin credentials:</p> <pre>http://STORAGE·DOMAIN:SCSP·BINDPORT/_admin/portal http://STORAGE·DOMAIN:80/_admin/portal (default)</pre> <p>The Content UI opens to the domain's page, from which the tenant's information can be accessed (if allowed) by clicking the breadcrumb menu.</p> 

Different Tenant or Domain	<p>To log in to a <i>different tenant or domain</i> with the existing credentials, specify the tenant or domain after the user name:</p> <ul style="list-style-type: none"> • Tenant – append a plus '+' sign and the tenant name: <code>username+rtenant</code> • Domain – append an at '@' sign and the domain name: <code>username@domain.com</code>
-----------------------------------	---

Creating a tenant

1. From the list of **Tenants**, click the **Add** button.
2. Type in the name of the new tenant. (See [Naming Rules for Swarm.](#))
3. Press Enter or click the **Add** button again to save it.

Note

The **SYSTEM TENANT** has no owner or configuration options; it is the permanent system-created tenant that manages any storage domain not associated with another tenant.

See [Configuring Tenants.](#)

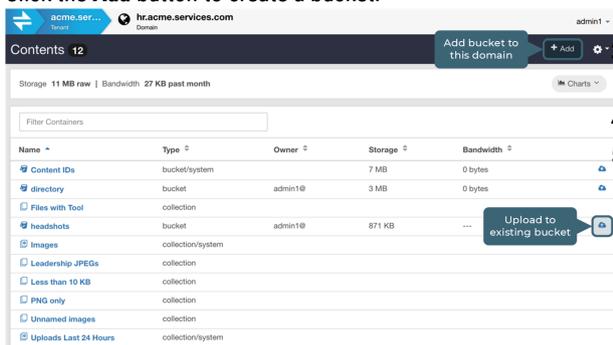
Creating a storage domain

1. Click on a tenant to view the list of domains within it.
2. Click the **Add** button to create a domain.
3. Type in the name of the new domain. (See [Naming Rules for Swarm.](#))
4. Select **Add** to save it.

See [Configuring Domains.](#)

Creating a bucket

1. Click on a domain to view the contents within it.
2. Click the **Add** button to create a bucket.



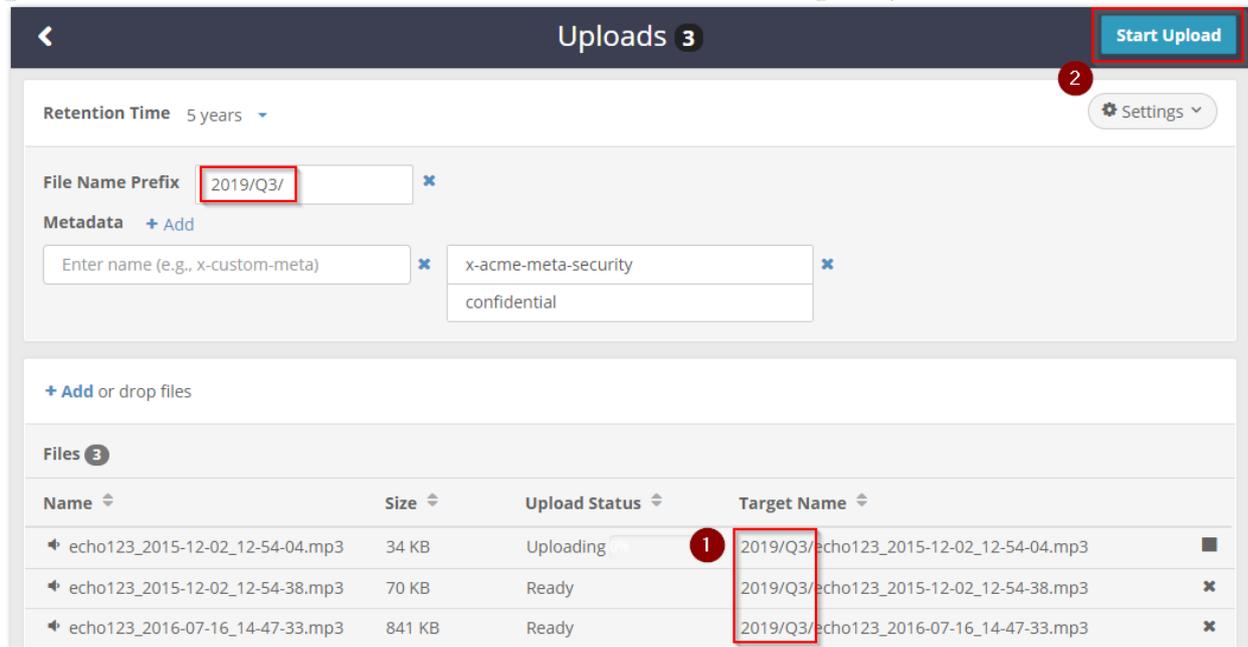
3. Clear the **S3 Compatible** checkbox if a non-S3-compliant name needs to be used.
4. Type in the name of the new bucket. (See [Naming Rules for Swarm.](#))
5. Select **Add** to save it.

See [Configuring Buckets.](#)

Uploading content

1. Click the **Uploads** button from the bucket view.
2. Set the **Retention time** if the default is not desired (*Keep until deleted*).
3. (optional) Click **Settings** to set custom options for file naming and metadata tagging:

4. Click **Add** to browse to local files or drag and drop them directly onto the upload area to queue them for upload:



The screenshot shows the 'Uploads' interface with the following details:

- Retention Time:** 5 years
- File Name Prefix:** 2019/Q3/
- Metadata:** x-acme-meta-security, confidential
- Files Table:**

Name	Size	Upload Status	Target Name
echo123_2015-12-02_12-54-04.mp3	34 KB	Uploading	2019/Q3/echo123_2015-12-02_12-54-04.mp3
echo123_2015-12-02_12-54-38.mp3	70 KB	Ready	2019/Q3/echo123_2015-12-02_12-54-38.mp3
echo123_2016-07-16_14-47-33.mp3	841 KB	Ready	2019/Q3/echo123_2016-07-16_14-47-33.mp3

Tip
Check the **Target Name** column to validate the final object names before you start the upload. There is no name prefix option available when uploading files as Content IDs.

5. Click the **Start Upload** button to launch the upload with these settings.

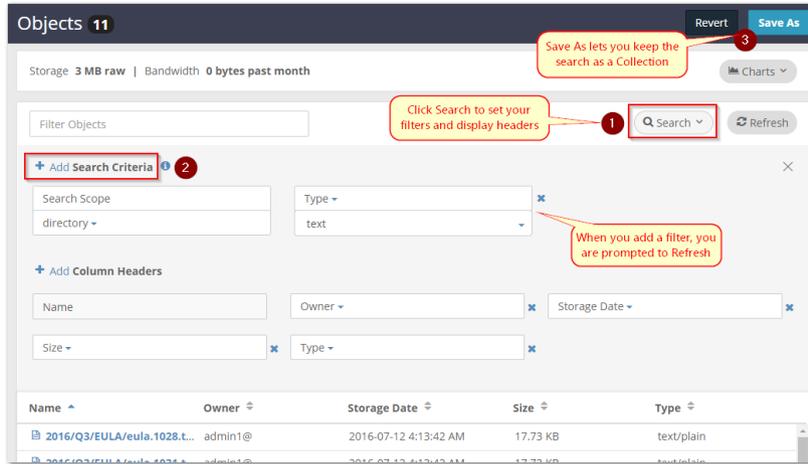
See [Uploading Files](#).

Creating a search collection

A new search is defined and run when a collection is added. The results can be saved as a named collection for future use after viewing the results.

1. Click on the domain name in the breadcrumbs bar to return to the domain.
2. Click the **Search** button to create a collection (which is a saved search).
3. Search for some of the uploaded data and click the Refresh button to rerun the search:

- **Filter Objects** - for string searches on object names.
- **Search Scope** - search the entire domain, unnamed objects (*Content IDs*), or a specific bucket.
- **+ Add Search Criteria** - specify a new search filter, such as **Type** equal to the string `text`.
- **+ Add Column Headers** - customize the view the list of matched content on the bottom half of the screen.

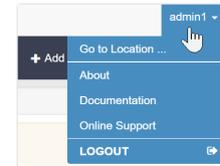


4. Click the **Save As** button at top right to save the search and display as a collection for future use.

See [Search Collections](#).

Resources

Located at the top right of the Content UI is the account name, which drops down a menu of resources:



Go to Location ... – opens a window for quick navigation to resources by name
About – reports the version of the software

in use

- **Documentation** – opens searchable online help
- **Online Support** – opens the [Support](#) site
- **Logout** – ends the current session

See [Go to Location](#).

Naming Rules for Swarm

- [Slashes](#)
- [Note](#)

Follow these rules for naming the domain, bucket, and named objects created for storage in Swarm.

Slashes

Swarm handles slashes this way (v11.1):

- Leading slashes (/foo) are silently removed in all cases.
- Trailing slashes (foo/) are silently removed for buckets, but they cause *404 Page Not Found* errors for domains.
- Trailing slashes (foo/bar/) are preserved for object names because they are valid.

Type	Reference	Rules and Notes	Examples
Tenant	RFC 1034	<p><i>Applies to Gateway only.</i></p> <p>A tenant must follow the naming rules of a domain.</p>	
Domain	RFC 1034	<p>For maximum compatibility, verify the domains are valid DNS names that resolve in your network.</p> <p>A domain name must</p> <ul style="list-style-type: none"> • Be a 7-bit ASCII byte sequence. • Be case-insensitive. • Begin with an alphanumeric character. • Use alphanumeric characters, underscore (_), period (.), and hyphen (-). • Not have adjacent or final hyphens or periods (-, .., -. , -). • Not be an IPv4 or IPv6 IP address. • (<i>S3 compatibility</i>) Not be longer than 253 characters. 	<p>Valid:</p> <pre>my-cluster.example.com my_cluster.example.com</pre> <p>Invalid:</p> <pre>domain cluster_example_com</pre>
Bucket	RFC 1034	<p>A bucket name (which is only used in the path) must</p> <ul style="list-style-type: none"> • Be unique within the domain. • Be case-sensitive. • Be a valid URL-encoded, UTF-8 byte sequence. <ul style="list-style-type: none"> • <i>Content UI:</i> URL encoding is managed by the user interface. • Not be a UUID (32 hexadecimal characters). • Not exceed 8000 characters (greater than that is not tested or supported). • (<i>S3 compatibility</i>) Use lowercase ASCII and DNS-compatible names not longer than 63 characters. 	

Named object	RFC 3986	An object name must <ul style="list-style-type: none"> • Be unique within the bucket. • Be case-sensitive. • Be a valid URL-encoded, UTF-8 byte sequence. <ul style="list-style-type: none"> • <i>Content UI</i>: URL encoding is managed by the user interface. 	Valid: <code>Accounting/</code> <code>Customer23-03/15</code>
---------------------	--------------------------	---	--


Note

While you may use non-ASCII characters (such as "résumé.doc") in bucket and object names, the URL must be properly escaped in the HTTP request ("r%C3%A9sum%C3%A9.doc").

Go to Location

The Content UI includes a quick navigation feature, **Go to Location**, which is accessed from the global resources menu that drops from the log in:

This window allows jumping directly to a specific resource (tenant, domain, or bucket) within the cluster:

i

Case sensitivity

Bucket names *are* case-sensitive.

Tenants and domains *are not* case-sensitive; for S3 compatibility, use lower-case, with the exception of the special SYSTEM TENANT.

The jump option displays from an error message, redirecting from a location not found:

i

Tip

Update the [collections](#) (stored searches) that reference a bucket if the bucket name changes: select a new Search Scope, update the bucket reference, or remove the collection if no longer valid.

Tenant
Domain
Bucket

Search Collections

- [Collection Essentials](#)
 - [System-Created Collections](#)
 - [Tip](#)
 - [Tip](#)
 - [New Search by Name](#)
- [Using the Search Panel](#)
 - [Context dependence](#)
 - [Note](#)
 - [Setting Search Criteria](#)
 - [Search on Common Metadata](#)
 - [Searching Extended and Custom Metadata](#)
 - [Example Metadata Search](#)
 - [Tip](#)
- [Searching by Metadata Selected from Objects](#)

Collection Essentials

A *collection* is the result set of a search run against a domain (or bucket) and then saved. Collections are listed among the buckets at the domain level:

Name	Type
Content IDs	bucket/system
directory	bucket
Files with Tool	collection
headshots	bucket
Images	collection/system

Collections shape the view of data in three ways:

1. **Scope** – Set the *scope* of the search (either an entire domain or a bucket)
2. **Filters** – Add *search criteria* for filtering (by name, owner, size, type, date, and/or metadata)
3. **Display** – Add/remove *columns* to display (such as to add metadata or custom metadata fields to the view)

System-Created Collections

The **Contents** view appears when opening a domain provides quick access to the buckets, files, and upload activity of the domain. Domains include five permanent system collections for common domain-wide inquiries by default:

- *Images* – domain-wide listing of all files of **Type image**, across all buckets
- *Uploads Last 24 Hours* – all files uploaded in the last 24 hours, across all buckets
- *Uploads Last 30 Days* – all files uploaded in the last 30 days, across all buckets
- *Uploads Last 7 Days* – all files uploaded in the last 7 days, across all buckets

Tip

Identify the default collections by **Type, collection/system**.

Name	Type	Owner	Storage	Bandwidth
Content IDs	bucket/system		0 bytes	5 KB
directory	bucket	admin1@	2 MB	841 KB
headshots	bucket	admin1@	---	---
Images	collection/system			

The **Images** collection lists *all* uploaded graphic files of **Type "image"**, both named and unnamed (UUID), across all named buckets as well as the **Content IDs** bucket:

Tip

Tip

Each of the columns allows ascending and descending sorting. Click the column header to toggle the sort direction.

Name	Bucket	Type	Size	Owner	Storage Date
d9beebd43c7e39b7df5f27c123329dcd	Content.IDs	image/jpeg	546.08 KB	admin1@	2016-07-14 9:26:27 PM
e5cd0a290daa6fe8ae9bb2df6f088be	Content.IDs	image/png	585 bytes	admin1@	2016-07-13 7:05:11 PM
fc15cf0719eec63872e256992ed12336	Content.IDs	image/jpeg	520.63 KB	admin1@	2016-07-14 9:26:26 PM
icon-release-notes.png	directory	image/png	8.81 KB	admin1@	2016-07-11 9:25:26 PM

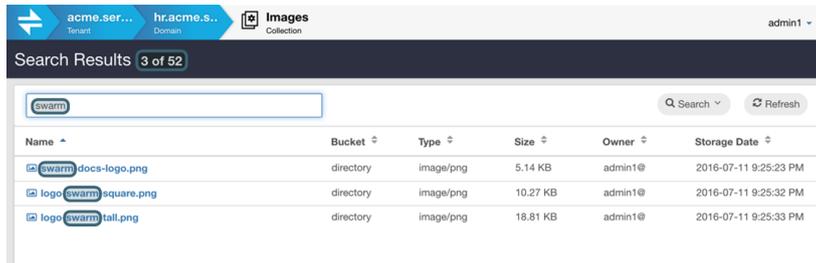
Narrow the listing by entering a string in the box filtering by **Name** to find a target faster.

Although these system-generated collections cannot be changed or deleted, build on this set with new collections

(saved searches) of custom design. Save a permanent collection under a new name using the **Save As** button if modified.

New Search by Name

Enter a string (no wildcards) in the **Filter** box when narrowing down by name (as performed in file system searches) when a large number of objects exist. The case-insensitive search begins as typing begins: enter "b" and any object containing letter B (regardless of case) somewhere in the name appears in the list.

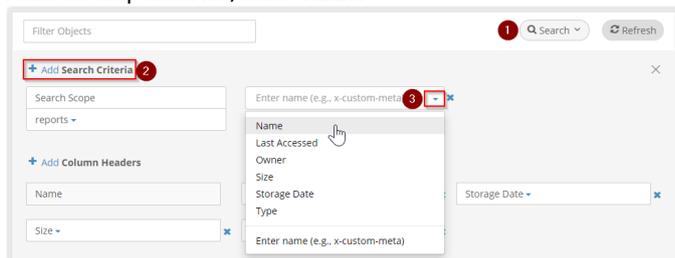


The **Objects** count shows what portion of the total listing matches the string (here, 3 of 7).

Use the full **Search** controls if this is a search to be repeated or make available for other users:

1. Delete the string in the Filter Objects box and click the **Search** button to open the search pane.
2. Click **+ Add Search Criteria**, which adds a new (empty) criteria operation.

3. From the drop-down list, select **Name**.



4. Enter the string used in the Filter Objects box when prompted for the **Value** to match on Name.
5. The **Refresh** button flashes to prompt to run the search. Click **Refresh** to verify the search returns the same objects as before.
6. At top, click **Save As**, and name the new collection.

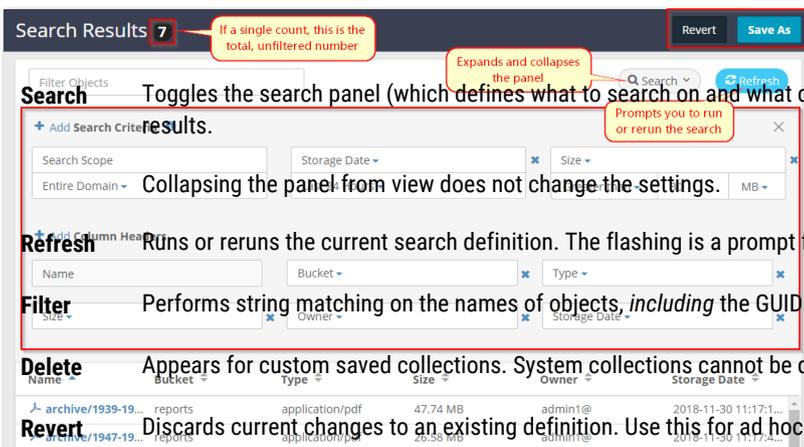
Using the Search Panel

The Collections feature allows performing complex ad hoc searches and define custom saved searches and views.

Context dependence

Collections depend on the existence of containers (bucket, domain). Edit the collection to update it if the bucket was renamed or recreated. Update or delete the collection if the original container was deleted.

In the listing for every collection, the **Search Results** panel appears with a results counter:



The search commands have these effects:

Search Toggles the search panel (which defines what to search on and what columns to return) in and out of view, above the search results.

Refresh Runs or reruns the current search definition. The flashing is a prompt for to rerun the search because changes are detected.

Filter Performs string matching on the names of objects, including the GUIDs of unnamed objects.

Delete Appears for custom saved collections. System collections cannot be deleted.

Revert Discards current changes to an existing definition. Use this for ad hoc searches, to avoid keeping unneeded collections.

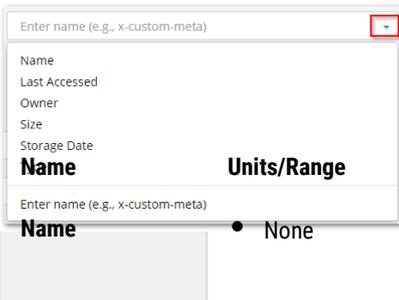
Save As Saves current definition (scope, criteria, and columns) for later use.

Note

Follow the [Naming Rules for Swarm](#) for named objects and verify it is unique to the domain when creating a collection.

Setting Search Criteria

Define search criteria against basic metadata, extended metadata, and custom metadata. Click the **+Add** button as many times as needed to combine search criteria to narrow the results to the desired data.



Search on Common Metadata

Several commonly searched attributes are predefined for ready access:

Name	Notes
	Use wildcards to specify the string to match on: *cert*

Last Accessed	<ul style="list-style-type: none"> • Last 24 Hours • Last 7 Days • Last 30 Days • Last (custom) • Before... • Since... 	<p><i>For use only if the cluster is storing this information; this feature must be enabled via the Swarm Storage setting for Time of Last Access - atime. (v11.0)</i></p> <p>Shows the value in the Castor-System-Accessed header, which is indexed in Elasticsearch as 'accessed'.</p>
Owner	<ul style="list-style-type: none"> • None 	<p>Use wildcards to specify the string to match on:</p> <p><code>*admin*</code></p>
Size	<ul style="list-style-type: none"> • bytes • KB - kilobytes • MB - megabytes • GB - gigabytes • TB - terabytes • PB - petabytes • EB - exabytes 	<p>Select the operation for the comparison:</p> <ul style="list-style-type: none"> • Greater than • Equals • Less than
Storage Date	<ul style="list-style-type: none"> • Last 24 Hours • Last 7 Days • Last 30 Days • Last (custom) • Before... • Since... 	
Type	<ul style="list-style-type: none"> • Audio • Image • PDF • Text • Video • None • Enter value (such as <code>image/jpeg</code>) 	

Searching Extended and Custom Metadata

Search against any system or custom metadata stored with each object. The following shows common metadata included on an object's detail view:

Metadata	Example value	Notes
Size	117.12 KB	
Type	image/jpeg	
Owner	admin1@	
Stored Date	2015-09-23 5:57:25 PM	

Castor-System-Cid	7da76343ad6bc9f2f739f0595a2756e4	
Castor-System-Cluster	raindance	
Castor-System-Created	2015-09-23 5:57:25 PM	
Castor-System-Name	jsmith.jpg	
Castor-System-Version	1443049045.780	
Content-Disposition	attachment; filename="jsmith.jpg"	Stores the original name of the uploaded source file .
Content-Md5	5QET59jX1t8//iD4CgnWWQ==	
Etag	"9dbfd0d4b524e8914280b0b1f7d12e3b"	
Lifepoint	[Tue, 29 Sep 2015 05:00:00 GMT] deletable, [] delete	Stores the lifepoint settings in force for the object, if any exist. The example shows the object was imported with a specific expiration date.
X-Last-Modified-By-Meta	admin1@	
X-<custom-tag-name>-Meta	2008-01-15 12:00:00 AM	Custom metadata tags entered when the file was stored. The example shows hire date data corresponding to this custom tag: <code>X-Hiredate-Meta</code>

Example Metadata Search



Tip

Specify whitespace value to search for results have the metadata tag but with no associated value. Leading or trailing white space in text strings for names or metadata tags are ignored.

Suppose a set of files are uploaded into the **Content IDs** bucket, so they are stored by UUID. The original filenames are stored as metadata. Take these steps if a view of the source filename for each image is desired:

1. From the domain list, open the **Content IDs** bucket.
2. In the **Column Headers** section, click **+Add**.
3. Type in the name of the metadata field storing the source filename: **Content-Disposition**.
4. Click Refresh to populate the new column:

Column Headers

Size x Type x Content-Disposition x

+ Add

Name	Size	Type	Content-Disposition
94dbd6fbab5542300a8c9c018...	1.01 KB	image/png	inline, filename="UTF-8" 'Icons.InvertColorsEffect.png
616eee0840546cd3fe7a6e926...	843 bytes	text/plain	inline, filename="UTF-8" 'copying.txt

5. Create a **Search Criteria** on **Content-Disposition**, then Refresh to narrow the results to files containing "Tool" in the name originally:

Search Criteria

Search Scope

Content IDs

Content-Disposition x Enter name (e.g., x-custom-meta) x

Tool

+ Add

Column Headers

Size x Type x Content-Disposition x

+ Add

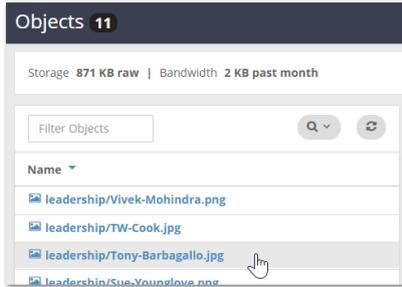
Name	Size	Type	Content-Disposition
bc49abf9b236afdefbd57c83ec...	661 bytes	image/png	inline, filename="UTF-8" 'Icons.Settings Tools .24.png
899f5a07cb7688ec6389eee3d...	520 bytes	image/png	inline, filename="UTF-8" 'Icons.MenuWindow Tools con.png

6. Select **Save As** and provide a name for the collection for future reference to keep this collection; otherwise, click **Revert**.

Searching by Metadata Selected from Objects

There is tremendous utility in building search collections based on metadata, especially the extended metadata Swarm indexes and custom metadata. The easiest method for searching this kind of metadata is to start from an object containing the desired metadata.

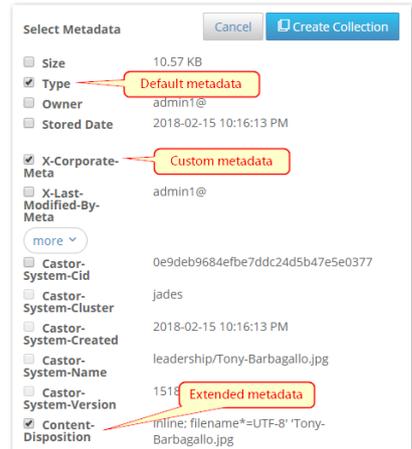
1. Find and double-click the object to view the details:



2. Click **Create Collection** in the detail view:

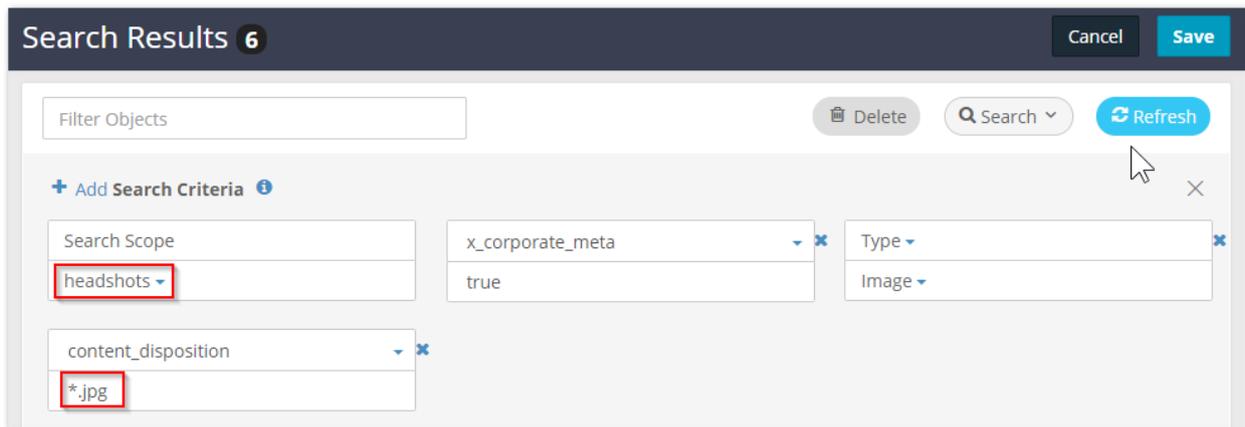


3. Select the desired fields, then select **Create Collection**:



4. Edit

Search Criteria in the **Search Results**, and click **Refresh** to test the results:



Tip

Lists are truncated to 10,000 objects to keep the visual display of lists manageable. Apply more filtering to return a shorter list.

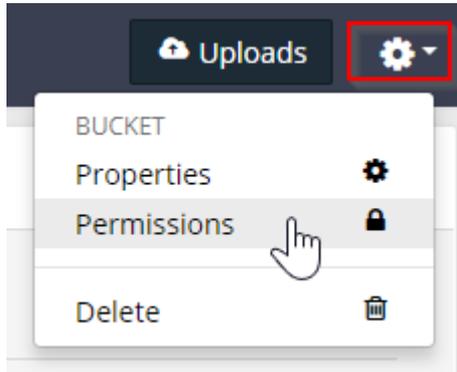
5. Improve the listing display by changing the **Column Headers** as needed (move, add, delete) when the filtering returns the correct results.



Setting Permissions

- [Tip](#)
- [Important](#)

Permissions are determined by the active ACL (*access control list*) policy, which is a list of rules that grant or deny users and groups the ability to perform specific actions.



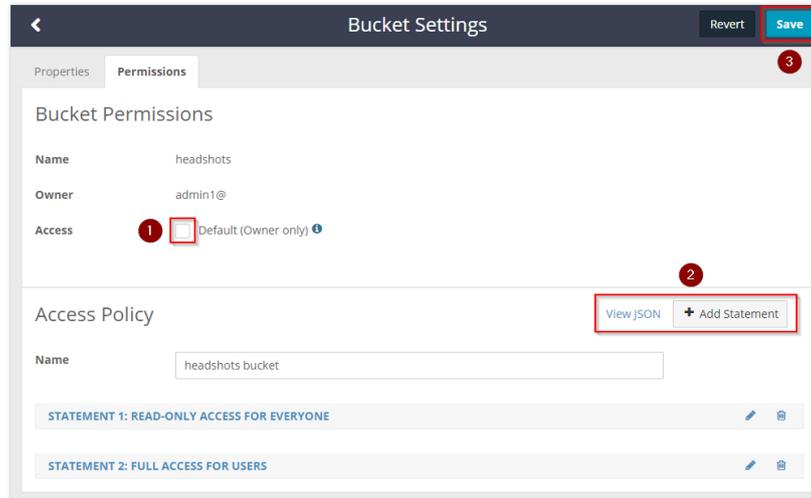
Default (Owner only) access applies automatically, referring to the owner of the current tenant, domain, or bucket. The owner has access *unless* a parent scope grants additional permissions to other users and groups in the absence of an access control policy.

Name	headshots
Owner	admin1@
Access	<input type="checkbox"/> Default (Owner only) ⓘ

See [Gateway Access Control Policies](#) for the usage and

components of policies.

An interactive policy editor expands so a policy for the current tenant, domain, or bucket can be created when unchecking **Default (Owner only)**. (v9.4)



The editor includes templates for adding the most commonly needed policies (such as *public read-only* and *authorized user full access*) as well as options for designing granular access for users and groups. Safeguards help protect from unintended consequences, such as denying access to *All Authorized Users*, which has the effect of locking out the Owner as well.

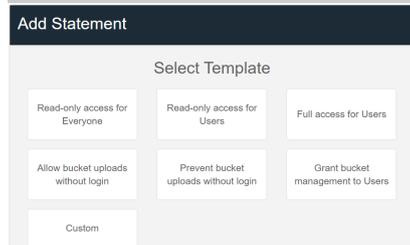
Note these behaviors and cautions:

Add statement

From the **+ Add Statement** dialog, copy existing definitions to alter (changes do not affect the originals). Select the template that is closest to the desired policy, and edit it for any needs.

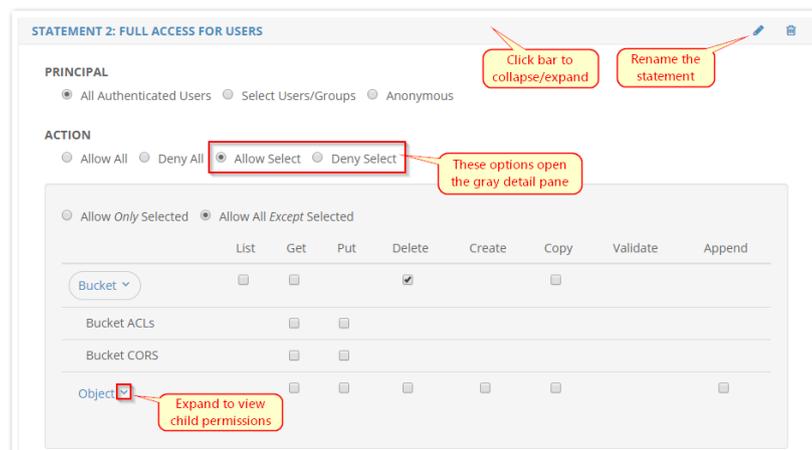
Tip

Rename the default statement name to describe the new effect: click the **Edit** (pencil) icon in the statement's title bar.



View statement

The title bar of each statement is a toggle: click it to expand and hide the statement settings



Undo edits

Select **Revert** to undo any *unsaved* policy changes made. This clears any changes that are pending.

Delete policy

Click the **Delete** (trashcan) icon in the title bar to remove a single statement. The change takes effect when clicking **Save**.

Re-enable **Default (Owner only)** and select **Save** to remove the entire *existing* access policy. *Use caution as this cannot be undone.*

Statement counter

The counter prefix the editor adds to statement names verifies each statement name is unique; additional inspection is recommended if removing these in the JSON editing view.



View JSON

Select **View JSON** to view (and optionally edit) the underlying JSON; select **Hide JSON** to return to the interactive editor, unless changes prohibit the use (*see next*).

Important

In the JSON view, "Version" specifies the version of the [AWS policy language](#), not the policy's contents. "Version" must be set to the current (2012-10-17) or prior (2008-10-17) language version, or else JSON validation errors prevent saving.

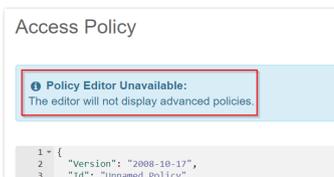
Read the modification time and author that are stored as standard metadata on the relevant policy.json object if needing to find the date and user responsible for the last policy update.

Advanced policies

The interactive editor does not open for *advanced* policies, which are those that involve these complexities:

- **Resource** – Anything other than asterisk (*) or all
- **Principal** – Either:
 - Is AWS
 - Has any conditions (such as match criteria) with or without child properties

Advanced policies can be viewed and edited through the JSON view. The interactive editor is disabled and the JSON is edited directly if the existing policy has the above elements:

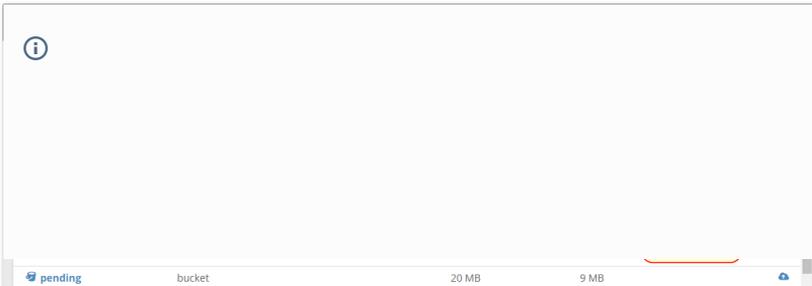


Prefixes

Prefixes are deprecated, negatively impact performance, and are ignored by the policy evaluation. The interactive editor has validation to remove them, which verifies policies work as expected. This includes the following: `ldap`, `pam`, `arn`, `aws`, `s3`

Uploading Files

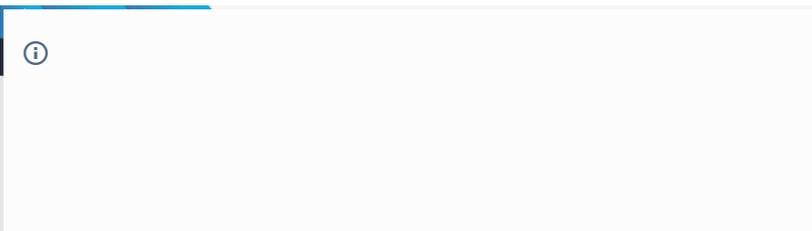
Although the bulk of each tenant's content is likely to be uploaded by applications that integrate with Swarm, you can view and upload files directly from the Content UI.



Tip

The **Content IDs** bucket is a system-generated container that manages all unnamed objects, which are identified by the UUID. Files are stored with new UUIDs with the source file name saved as the **Content-Disposition** metadata value if uploaded to this bucket.

When uploading files, a **Retention Time** is required to be set, but the **Settings** panel can be expanded, which allows additional control over the upload. Complete the options in this order:



Tip

Set the upload options carefully and verify the target names *before* selecting **Start Upload**. It is faster to delete files from the upload queue than to clear out objects created erroneously.

Setting	Values	Example	Notes
Retention time	<ul style="list-style-type: none"> Keep until deleted 1 year 3 years 5 years Date selector 	3 years	Required. Defaults to Keep until deleted , which retains the object indefinitely. Select a preset duration or pick a date after which the files are automatically deleted by Swarm.

Note

The uploaded content does not have a Lifepoint defined in its metadata if the default is kept and no expiration is set.

File Name Prefix	{string}	2016/Q3/West/
-------------------------	----------	---------------

(optional) Enter the prefix to add to the name of each file being uploaded. Use slashes in the prefix to help organize the content, but performing this does not *create* buckets or folders within buckets: the prefix becomes part of the content's name in Swarm. No trailing '/' is appended to the prefix, so a prefix of "2016/Q3/West" and a file name of "filename.txt" results in an object name of "2016/Q3/Westfilename.txt".

Note

The name prefix option is not available when uploading to the Content IDs bucket because the files are assigned a UUID.

<p>Metadata</p>	<pre>{custom-header} = {custom-value}</pre>	<pre>x-status- meta = active x-public- meta = true</pre>	<p>(optional) Add one or more tags to the uploaded content. Metadata is stored with every object and may be used for Collections and searching within Swarm. Metadata names must match one of these patterns:</p> <ul style="list-style-type: none"> • x-*-meta • x-*-meta-* <div data-bbox="751 422 1484 583" style="border: 1px solid #ccc; padding: 5px;"> <p>Tip</p> <p>See Search Collections for how to make use of your custom metadata in searching and collections.</p> </div>
------------------------	---	---	--

After you start the upload, the status of each file upload is displayed dynamically:



Tip

You can cancel an individual file upload in progress by clicking the **x** icon at the end of the line.

Setting Quotas

- [Quota Essentials](#)
- [Enabling Quotas](#)
- [Defining a Quota](#)
- [Quota Effects](#)
- [Quota Headers](#)
- [API for Quotas](#)
- [Example Quota Scenarios](#)

Quota Essentials

The Gateway manages all quota actions; as with Metering, there is no additional service for installation and monitoring. Use the Content UI to set quotas on a context, as content protection and versioning are set. A quota policy combines these elements:

1. **Scope** - *where* the quota applies: a specific bucket, a domain, or an entire tenant
2. **Limits** - *how much* storage and/or network usage is allowed
3. **Action** - *what happens* in response overage, from notification to complete lockout
4. **Notification** - *who to email* with information about the quota status change
5. **Override** - (optional) *alternate action* in response of overage with an expiration time

Quota processing relies on periodic queries of Elasticsearch where historical bandwidth and storage metering is maintained. Gateway continually evaluates the metered data against the configured limits on tenants, domains, and buckets to determine when limits are exceeded. The action and optional override are used to determine the restrictions imposed while a quota remains above the limit.

During quota evaluation, if multiple limits are exceeded, including limits from different scopes within the hierarchy, then the most restrictive action or override are applied. See *Example Quota Scenarios* below for examples of multiple limits interacting within the scope hierarchy.

An expiration date must be provided when an administrator uses an override to force an action different from the computed one. Overrides are designed as a temporary measure to alter the effects of exceeding a quota limit without requiring the action to be changed. The override is automatically invalidated by the Gateway so an administrator does not need to remove any overrides put in place upon reaching the expiration date. The quota's defined action again applies if the limit is exceeded once an override expires.

Purpose for Quotas

No quotas exist by default. All users are entitled to as much network bandwidth and storage space as the storage cluster allows. Design quota policies using metrics to enforce service level agreements to create a cloud service managing and billing tenants. Scope owners may want to define self-imposed quota limits, such as to put a safety cap on the overall usage to control the bill.

Quota Metrics

Quota policies monitor and allow limiting two classes of metrics: bandwidth usage and/or storage usage:

- **Bandwidth**: total of network bytes IN plus bytes OUT. Bytes IN refers to data sent from the client application to the front-side interface of the Gateway. Bytes OUT refers to data sent from the Gateway to the client application. These measures include the content body of the HTTP requests. Bandwidth from replication feeds are included when a feed is routed through the Gateway. Bandwidth from Management API requests are not included.
- **Storage in use** is one of two types: Raw and Logical. Raw storage refers to the actual amount of disk space used within the cluster based on the replication/EC factor for the objects. Logical storage refers to the summation of the objects' Content-Length headers. One type is allowed per quota policy.

Cascading Limits

Assign quotas to one or more contexts: tenants, domains, and buckets. Swarm applies quota states in this order: tenants, then domains, then buckets. A bucket's quota is capped by the limits for the domain and tenant, and a domain's quota is capped by the limits for the tenant. Define quota limits that, when combined across a scope, exceed the limits allowed for the scope (such as 12 domains with 1-TB limits being housed in a tenant with a 10-TB limit). This flexibility simplifies the task of quota creation and makes it easier to manage quotas at lower levels.

Important

Exceeding a metric limit at a higher scope level cascades the overage across all lower scopes. All storage domains and all buckets are over quota, *regardless of the individual metrics* if a tenant's limits are exceeded. An empty bucket can be over quota because the domain is over quota.

Over-provisioning

They show the limits as specified for the scope level when metric limits are displayed in the Content Portal. The limits display as if they are unlimited if no quota exists at the scope level. The metric limit for a bucket with a quota displays the bucket's metric limits even if they exceed the domain's metric limits or the tenant's metric limits. This allows administrators to over-provision storage for lower scopes. The Content Portal displays those limits in the lower scopes as they are possible before the limits at the higher scopes are exceeded.

Enabling Quotas

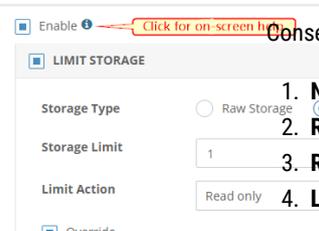
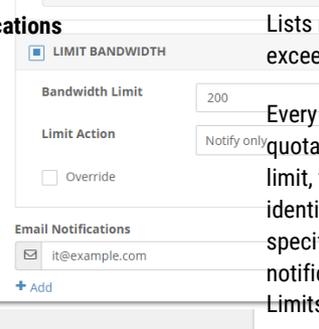
By default, the Quotas feature is disabled globally, and it must first be enabled through Gateway. The Gateway configuration has a `[quota]` section for enabling, controlling, and customizing notifications for quotas.

See [Gateway Configuration](#).

Defining a Quota

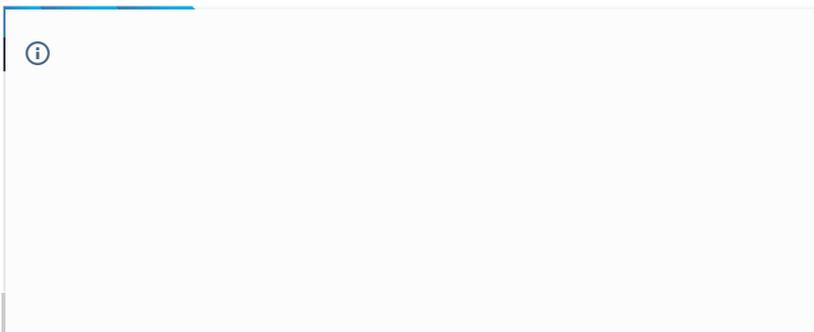
Navigate to the **Properties** (gear icon) of the tenant, domain, or bucket to define a quota. Click **Enable** to view the policy settings:

Enable	The feature must be deliberately enabled for this scope, after which all required values must be entered to Save .
Storage Type	Choose to monitor storage usage by <ul style="list-style-type: none"> • Raw Storage (the total footprint on disk of all objects, replicas, segments, manifests) • Logical Storage (the sum of the uploaded content and any versions).
Storage/Bandwidth Limit	Limits are set in units of MB, GB, TB, or PB. <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <h3> Note</h3> <p>Bandwidth resets automatically based upon a calendar month: the first day of the month at 00:00:00 UTC. The status automatically changes to OK when a metric falls back within the allowed limit or resets for the new month.</p> </div>

<p>Actions</p>		<p>Consequences for exceeding a quota, from least restrictive to most:</p> <ol style="list-style-type: none"> 1. Notify only: Sends notification but does not restrict any operations 2. Read/Delete only: Blocks operations that may increase storage (no writes, updates) 3. Read only: Blocks operations that change or add content (no writes, updates, deletes) 4. Lockout: Blocks all operations
<p>Override</p>		<p>Sets an alternate Action with an Expiration date as a temporary policy override, such as to grant a grace period for the client to resolve the overage. Overrides may impose a more restrictive Action than the policy, such as may be used if a tenant is delinquent in payments.</p>
<p>Email Notifications</p>		<p>Lists recipients of email notifications, who are notified when a limit is exceeded or is no longer exceeded.</p> <p>Every time an overage begins or ends, the Gateway sends an email to every address defined within the quota policy's notification email list. These notifications contain the scope, the metric's name and limit, the time of detection, and the resulting quota state that now applies to the scope. The scope is identified as the tenant name, the domain name, or the domain + bucket name. Notifications are specific to the scope where the overage occurs. For example, if a tenant quota is exceeded, notifications do not cascade down to the people in the domain and bucket quotas within the tenant. Limits cascade to lower levels, but notifications do not.</p>

Quota Effects

The Content Portal displays an alert message about the overage on its **Contents** page and provides the scope in which it has occurred (the current level or higher) when a metric limit has been exceeded by a tenant, domain, or bucket. Note: the overage may not be the fault of the current context. A user of a domain or bucket within the tenant sees an alert message that quota levels are exceeded, even if the domain or bucket is using *none* of the allowed usage if a tenant has exceeded the storage quota:

	<p>Note</p> <p>Quota metrics are measured periodically, so there can be a lag between when a metric limit is actually exceeded and when the overage is detected. A similar lag can exist going the other direction, as an overage ends. Lag is not longer than 60 minutes while the typical lag is 5 to 15 minutes in a typical deployment.</p>
--	--

HTTP response for overage - The HTTP response indicates where the overage occurred, the metric exceeded, and the current quota state when a storage action fails due to an exceeded quota. For example, if a domain's storage limit is exceeded while a bucket's storage limit is not exceeded and the quota state is "read only," a write operation to the bucket returns an error stating the domain's storage quota is exceeded.

Quota Headers

Define quota limits on context objects: tenants, domains, and buckets. Quota management makes use of metadata headers stored with these context objects.

In the header names, **{M}** is one of these usage metrics:

- `bandwidth` (network bandwidth inbound and outbound)

- `rawstorage` (storage in use as the total footprint on disk)
- `storage` (storage in use as the total logical size only)

Header	Value	Description
<code>x-caringo-meta-quota-{M}-limit</code>	= {state}; {limit} <ul style="list-style-type: none"> • limit = target for this particular metric • state = "notify", "nowrite", "read", "lock" 	The configured quota limit and the Action (consequence) when the limit is exceeded.
<code>x-caringo-meta-quota-{M}-current</code>	= {state}; {value}; {timestamp} <ul style="list-style-type: none"> • value = current for this particular metric • state = "ok", "notify", "nowrite", "read", "lock" • timestamp = ISO-8601 timestamp of last update 	Last measured value of storage and bandwidth metrics as queried from Elasticsearch. This header is computed by Gateway and is recommended not be set externally.
<code>x-caringo-meta-quota-{M}-override</code>	= {state}; {user}; [{deadline}] <ul style="list-style-type: none"> • deadline = ISO-8601 timestamp of expiration • state = "ok", "notify", "nowrite", "read", "lock" • user = who set the override 	<i>Optional.</i> The state and expiration time for the temporary override.
<code>x-caringo-meta-quota-email</code>	= {addresses} <ul style="list-style-type: none"> • addresses = comma-separated list of email addresses 	One or more email addresses to notify about quota state changes Notifications are sent whe the current state of a metric changes at the context level (tenant, domain, or bucket). The message text is similar to the error messages returned to blocked storage operations. The Gateway Configuration has a [quota] section where the email service, sender, and content of the email notifications for quotas are customized.

API for Quotas

Set and clear quotas and check on quota states using the [Content Management API](#).



Note

On Management API context listings, additional headers related to quotas may appear in the listing. These may be ignored.

See [Methods for Quotas](#).

Example Quota Scenarios

Here are two detailed scenarios for quota enforcement actions and how they apply within the scope hierarchy of tenants, domains, and buckets.

```
Tenant alpha: storage quota 1.0PB, limit action "read/delete only"
  Domain alpha-one: no quota
    Bucket mike: bandwidth quota 100TB, limit action "locked"
  Domain alpha-two: no quota
    Bucket november: no quota
```

During use, the sum of the storage in the two domains for tenant **alpha** expand past 1.0 PB. Tenant **alpha**'s storage quota has now been exceeded and enters a "read/delete only" state. This restriction also propagates down and applies to domains **alpha-one** and **alpha-two** and all buckets. No one is allowed to add more content -- only read and delete operations.

Later, as reading operations continue on bucket **mike** within domain **alpha-one**, the bandwidth total passes 100 TB and bucket **mike**'s quota is exceeded and enters a "locked" state.

At this point, no activity is allowed for bucket **mike** since it is "locked" while the other domains and buckets within tenant **alpha** remain in a "read/delete only" state.

After the end of the month is reached and bucket **mike**'s bandwidth is reset to zero, bucket **mike** is no longer exceeding the bandwidth quota. Bucket **mike** returns to the inherited state of "read/delete only" because the tenant is still in a "read/delete only" state.

```
Tenant bravo: bandwidth quota 500GB, limit action "locked"
  Domain bravo-three: storage quota 2.0PB, limit action "read only"
    Bucket oscar: no quota
  Domain bravo-four: no quota
    Bucket papa: bandwidth quota 250GB, limit action "notify only"
```

During use, domain **bravo-three** exceeds the storage quota of 2.0 PB and enters a "read only" state. Because of inheritance, this also means bucket **oscar** is now read-only. Later, bucket **papa** within domain **bravo-four** exceeds the bandwidth limit of 250 GB and enters a "notify only" state. Because all actions are still allowed, bucket **papa** can continue to use additional bandwidth. As this continues, tenant **bravo** eventually exceeds the bandwidth total of 500 GB and enters and "locked" state. Due to inheritance, this locked state now applies to domains **bravo-three** and **bravo-four** and to the buckets. Thus, bucket **oscar** and **papa** are now locked because of the tenant quota limit.

A few frantic phone calls after this lockout happens, the administrator agrees to override tenant **bravo**'s bandwidth quota with "notify only" for the remainder of the month. Thus, tenant **bravo**'s effective state is now "notify only" and all storage operations are again allowed for the lower scopes except for bucket **oscar**. Bucket **oscar** remains in a "read only" state because it is still over the storage limit, and no override exists for the quota. Bucket **papa** remains in "notify only" state, which is the same coming from tenant **bravo**, thus, storage operations are restored even though bucket **papa** remains over the limit.

Setting Tokens

- [Token Essentials](#)
 - [Best practices](#)
- [Accessing Tokens](#)
- [Creating Tokens](#)
 - [Important](#)
 - [Best practice](#)
- [Managing Tokens](#)
 - [Caution](#)

Token Essentials

In addition to HTTP Basic authentication, Gateway allows configuring token-based authentication. Token-based authentication works in two steps:

1. Request a token, by using HTTP Basic authentication to perform a one-time authentication within the Management API or to a special URI path in the Storage API.
2. Submit this token on all subsequent requests as proof of the user's credentials.

Tokens have these characteristics:

- **Ownership.** They are always owned by the user who created them, except for tokens created by the token administrator.
- **Expiration.** They expire at a fixed time after creation; default is 24 hours.
- **S3 key.** They may contain an optional secret access key for use with the S3 protocol.
- **Deletion.** Both the owner and the token administrator can list and delete the owner's active tokens.

See [Token-Based Authentication](#).



Best practices

- Token behavior cannot be selectively restricted (such as to work for specific actions or in specific domains/buckets). Prevent sharing of tokens with untrusted users/clients, as with any credentials.
- Fully qualify the names of any token administrators (such as `caringoadmin@` or `caringoadmin+acmetenant`) defined in an [IDSYS document](#) to avoid ambiguity when multiple IDSYS are used.

Accessing Tokens

Tokens can be accessed under the gear icon, which appears in the title bar of all tenants and domains (not buckets):

Creating Tokens

The default owner and expiration date can be overridden, as well as choosing to enable the S3 Secret Key when creating a token manually (for the current [tenant](#) or [domain](#)):



Important

The S3 Secret Key for the token *must be* copied from the **Success** message before closing it: for security reasons, the S3 Secret Key is not displayed in the Content UI after this point.

Best practice

Delete the token and create a new one so security is not compromised if S3 Secret Key is lost.

Notice: Once you close this message or navigate away from this page, the S3 secret key will not be displayed again!

Token ID: e46506cb4f1b9a5dctf5dd4bb351fbc
S3 Secret Key: lvjMF17G52eVPRNrzsctfxm2Un4bxFu6nl2bn004
 Expiration Date: 2017-09-23
 Owner: admin1
 Description: Token for new admin

Owner *
admin1

Description
Token for new admin

Expiration Date *
1 year

S3 Secret Key *
lvjMF17G52eVPRNrzsctfxm2Un4bxFu6nl2bn004

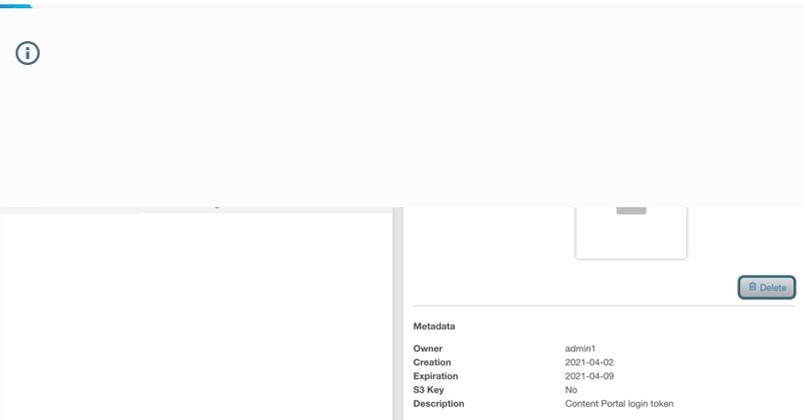
See [Integrating S3 Applications](#).

Managing Tokens

The UI lists *all* valid tokens, whether created here or programmatically, by the Management API. As soon as a token expires, it no longer appears in the listing and count of tokens.

Tokens are listed on the **Tokens** tab with a counter and a *Filter Tokens* field if any tokens exist for the particular tenant or domain, which allows searching for tokens matching the string within the **Owner** name or **Description** text. The S3 Secret Key is not displayed in the UI after creation for security reasons.

Double-click a token to view the properties and, optionally, delete it:

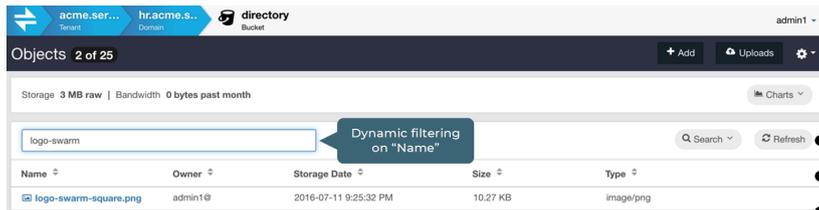


Caution

Tokens cannot be restored if deleted through this interface.

Content UI Overview

The Content UI offers a visual representation of the cluster, organized by tenants, domains, collections, and buckets. Depending on the level of the login credentials (**Root**, **Tenant**, or **Domain**) and the access policy in force, the Content UI displays only the information authorized; for example, domain-level users can at most view, add, or update buckets and bucket contents.



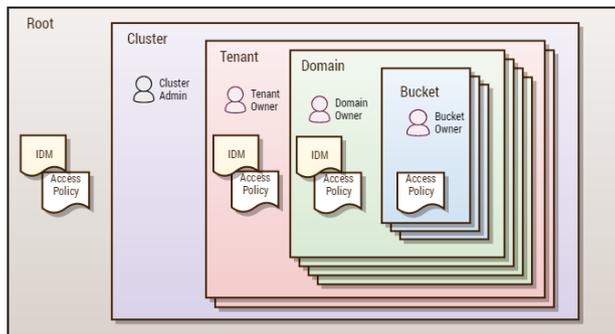
What it does – The Content UI layers on top of Gateway to provide a browser interface (which even your end users can access) for these key tasks:

- Creating tenants, domains, and buckets
- Uploading, downloading, listing, and sharing content
- Searching and filtering content by context (domain, bucket) and metadata (name, size, owner, type, date of

creation or last access, custom metadata)

- (Optional) Create new standalone [video clips](#) from uploaded videos (also known as *partial file restore*)

Controlling access – This diagram shows the hierarchy of the scopes (cluster, tenant, domain, bucket) across Gateway and your Swarm Storage cluster:



It is like nesting dolls: A *cluster* can contain multiple tenants; *tenants* can each contain multiple storage domains; *storage domains* are known to Swarm and are where content is stored.

Swarm defines the role of owner. Role-based access control (RBAC) definitions of varying complexity can be created as required for the organization using Gateway's access control policies. Note: the Cluster, Tenant, Domain, and Bucket "admins" shown in the diagram above are typical roles, but are not required or hard-coded into the system.

See [Content Gateway Concepts](#).

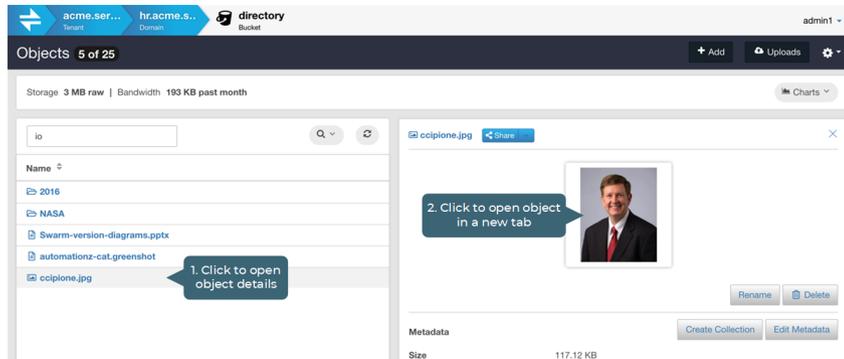
help-UIC

Downloading Content

- [Viewing Content](#)
- [Custom Metadata](#)
 - [Adding custom metadata](#)
 - [Searching custom metadata](#)
- [Deleting Content](#)
 - [Important](#)
- [Downloading Content](#)

Viewing Content

To access the details about an uploaded file, locate it in the listing and click on it. The details display at right:



Click the **more** button to see what advanced or Swarm-specific metadata was stored with the file:

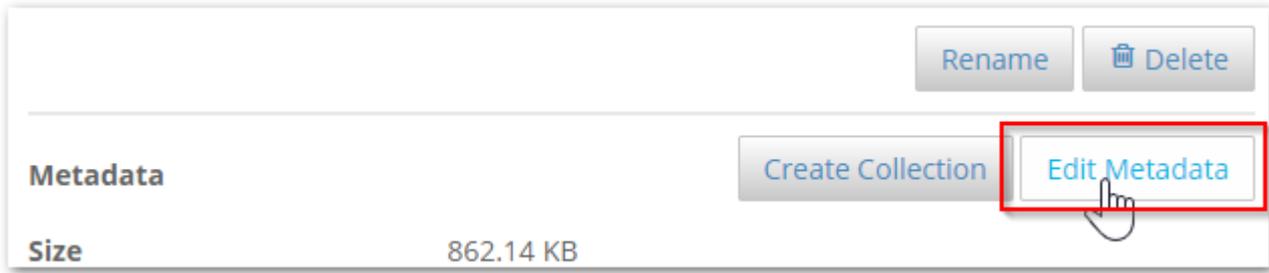
X-Last-Modified-By-Meta	admin1@
more	
Castor-System-Cid	babd0b717a7621c4413d800f6cec547a
Castor-System-Cluster	jades
Castor-System-Created	2016-09-23 9:52:53 PM
Castor-System-Name	ccipline.jpg
Castor-System-Version	1474667573.154
Content-Disposition	inline; filename*=UTF-8' 'ccipline.jpg
Content-Md5	5QET59jX1t8//ID4CgnWWQ==
Etag	"34e8003de10d7a02dc5541f12500b23a"

Custom Metadata

Any custom metadata stored with the file appears in this main view when the object details are opened:

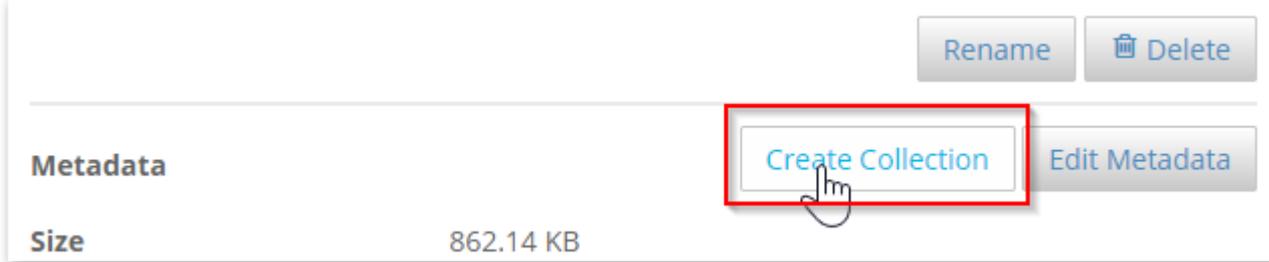
Adding custom metadata

Although custom metadata is typically added programmatically during ingest, correct and add to it using the **Edit Metadata** command:



Searching custom metadata

To search on metadata, use the **Create Collection** command:



Check the boxes for which metadata headers to have available to filter in the search:

Rename Delete

Select Metadata

Cancel Create Collection

<input type="checkbox"/> Size	862.14 KB
<input checked="" type="checkbox"/> Type	image/jpeg
<input type="checkbox"/> Owner	admin1@
<input checked="" type="checkbox"/> Stored Date	2018-12-09 9:58:49 PM
<input type="checkbox"/> X-Copyright-Guidelines-Meta	https://www.nasa.gov/multimedia/guidelines/index.html
<input type="checkbox"/> X-Dc-Meta-Date	2013-03-21
<input type="checkbox"/> X-Last-Modified-By-Meta	admin1@
<input checked="" type="checkbox"/> X-Subject-Category-Meta	aircraft

© DataCore Software Corporation. All Rights Reserved.

Page 898 of 1608

Deleting Content

Click the **Delete** button to delete the object, but proceed with caution as this action cannot be undone.



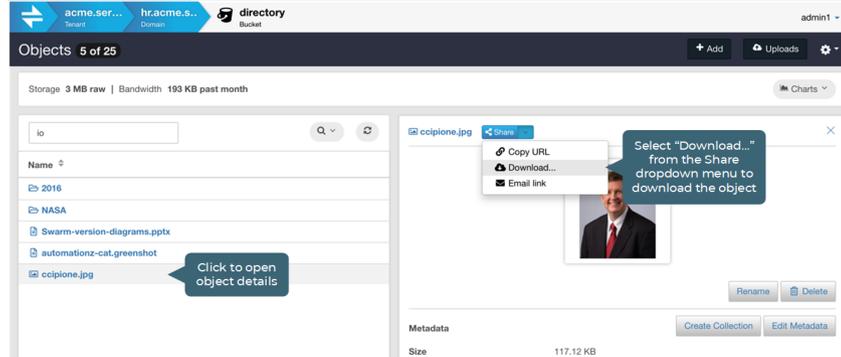
Important

Although deleted objects may continue to appear in Collection listings temporarily after they are deleted, they are no longer accessible in the storage cluster.

Downloading Content

Use browser commands to download a local copy of an object.

To view	In the collection or bucket listing, click the object name, then click on the thumbnail, at right.
To get link	Right-click on the object name and select Copy link address or Copy link location .
To download	Right-click on the object name and select Save link as...



Object Locking Content Portal

- [Object Locking Essentials](#)
 - [Retention periods](#)
 - [Retention modes](#)
 - [Legal hold](#)
- [Prerequisites](#)
- [Enabling Object Locking within a Bucket](#)
- [Applying Retention Locking](#)
 - [Inheriting Default Retention](#)
 - [Setting Up Retention on a New Object](#)
 - [Modifying Retention on an Existing Object Version](#)
- [Applying Legal Hold](#)
 - [Setting Up Legal Hold During Upload](#)
 - [Modifying Legal Hold on an Existing Object Version](#)

Object Locking Essentials

Object Locking prevents object versions from being deleted or overwritten for a fixed amount of time or indefinitely. It is applied to an object version to meet regulatory requirements that require WORM storage or to add another protection layer against object changes and deletion.

There is a strong connection between Object Locking and Versioning. Object Locking does not lock objects, but individual **object versions**. Therefore, a user can create new object versions even though the object is locked, but it is impossible to delete or change any locked version of the object.

There are two types of Object Locking:

- **Retention** - Specifies a fixed period ("retention period") during which the object version remains locked. During this retention period, the object is WORM-protected and cannot be overwritten or deleted. After the period expires, the lock goes away automatically from the object.
- **Legal hold** - Keeps the object locked until the legal hold is explicitly removed.

These two types of object locking are orthogonal, independent of each other, and can be used simultaneously.

Retention periods

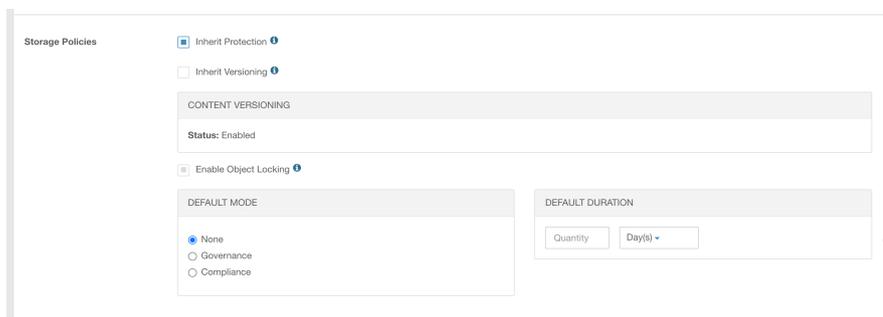
A **retention period** locks an object version for a fixed amount of time. Until that fixed amount of time has expired, one cannot delete or change the object version.

There are three different ways to set a retention period on an object version:

- Newly created objects can inherit a **default retention period** configured on the bucket.
- Explicitly set a retention period when creating a new object. This overrides the default retention period configured on the bucket if present.
- Explicitly set a retention period on an existing object version.

A bucket's default retention period specifies the duration in days or years, for which, every object version placed in the bucket is locked. While placing an object in the bucket, the Gateway calculates a retention period for the object version by adding the specified duration to the object version's creation timestamp.

An already set retention period can always be **extended**. Submit a new lock request for the object version with a retention period longer than the current period. The Gateway replaces the existing retention period with the new, longer period. Any user with permission to place an object retention period can extend a retention period.



Retention modes

There are two *retention modes* that impact what can be done with objects under retention:

- In **governance** mode, some users can be granted permission to shorten or remove a retention period if necessary.
- In **compliance** mode, any user including the admin

user, cannot overwrite or delete a protected object version. When an object is locked in the compliance mode, the retention mode cannot be changed, and the retention period cannot be shortened.

The default retention mode and retention period can be set independently at the bucket level. The retention mode always applies to the individual objects carrying it, not to the bucket or cluster as a whole.

Legal hold

A legal hold prevents an object version from being overwritten or deleted like a retention period. A legal hold does not have an associated retention period and remains in effect until removed.

Legal holds are independent of retention periods and retention modes. As long as the bucket contains an object that has Object Locking enabled, the user can place and remove legal holds regardless of whether the specified object version has a retention period set or not. Placing a legal hold on an object version does not affect the retention mode or retention period for that object version.

Version

2021-12-15 6:34:49 PM UTC ▾

Rename

Delete

Metadata

Actions ▾

Size	167.94 KB
Type	(none)
Owner	bguetzlaff@
Stored Date	2021-12-15 6:34:49 PM
X-Last-Modified-By-Meta	bguetzlaff@
Legal Hold	On
Object Lock Mode	Compliance
Object Lock Expiry	Fri, 14 Jan 2022 18:44:39 GMT
more ▾	

Important

A legal hold cannot be applied as a default at the bucket level.

Prerequisites

- Object Locking must be enabled within the cluster.
- At least one bucket must be available, and that bucket must have versioning enabled.

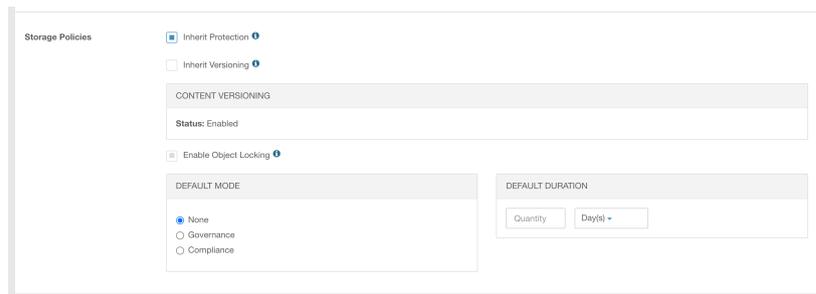
Enabling Object Locking within a Bucket

To apply Object Locking to an existing bucket, refer to the following steps:

Important

The bucket must have versioning enabled.

1. Search and select the target bucket name.
2. Click **Settings > Properties**.



3. Select the checkbox for *Enable Object Locking*.
4. To apply a default retention mode, select the appropriate default mode, either *Governance* or *Compliance*.
5. To apply a default retention duration, enter the default duration in days or years.
6. Click **Save**.

Once the configurations are saved, the bucket has Object Locking enabled. Any objects written to that bucket have the defined duration with the selected mode automatically applied, unless different values are provided at the time of write. The lock icon next to the bucket name represents that the bucket has Object Locking enabled. If the icon is green, then retention locking defaults are defined.

Images	collection/system
lots.of.stuff	bucket
object.locking.bucket	bucket
Uploads Last 24 Hours	collection/system

Note

Object Locking cannot be disabled once enabled for a bucket. The retention mode and duration can be updated.

Applying Retention Locking

A user can apply retention either via defaults set at the bucket level or any object versions of that bucket. There are three ways to apply retention on an object version:

- Create an object under the bucket that has Object Locking enabled. This newly created object inherits the default retention mode and period configured on the bucket.
- Explicitly set a retention period when creating a new object. This overrides the default retention period configured on the bucket if present.
- Explicitly set a retention period on an existing object version.

Info

Each object within a given bucket can have unique and independent retention policies applied to them.

Inheriting Default Retention

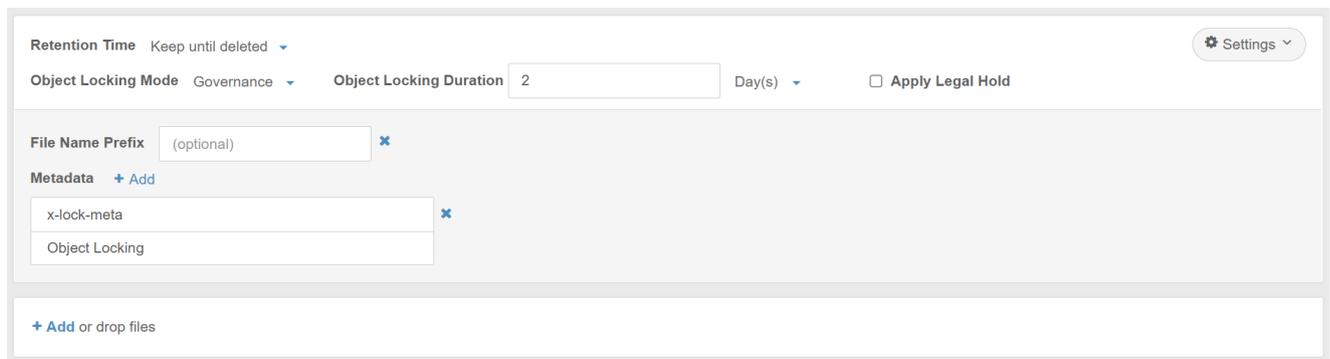
Before inheriting the default retention period of a bucket, verify the default retention mode and retention period under the *Bucket's Settings > Properties*. To learn more about uploading through Content UI, see [Uploading Files](#).

Info

When uploading an object via Content UI, any defaults defined at the bucket level are visible during the upload process. If either the mode or duration is missing, the upload process does not proceed until the values are provided for that specific upload.

New object versions can inherit the default retention configuration of the buckets by referring to the following steps:

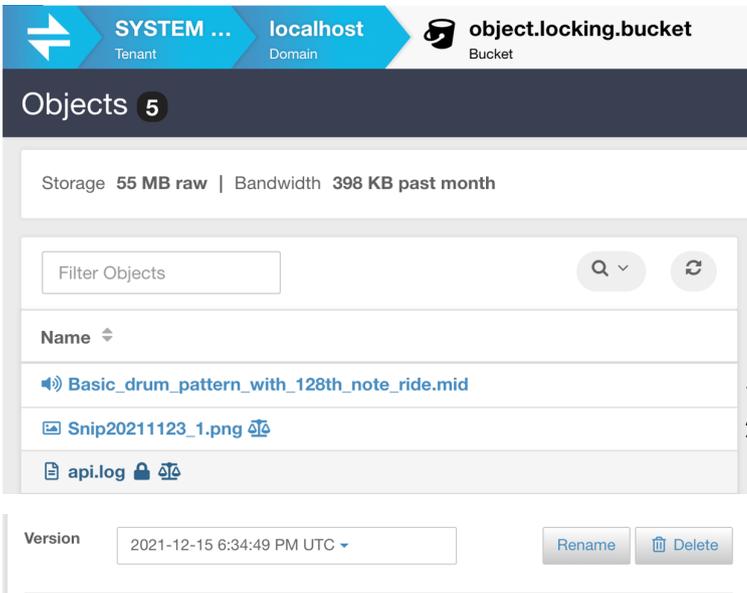
1. Navigate to the upload page for the target bucket in Content UI.
2. The upload settings for Object Locking are located under *Retention Time*. The defaults set at the bucket level are automatically applied. The upload does not proceed until all required values are present if either default is missing.



The screenshot shows a configuration panel for object upload. At the top, there are dropdown menus for 'Retention Time' (set to 'Keep until deleted') and 'Object Locking Mode' (set to 'Governance'). Next to 'Object Locking Mode' is a text input for 'Object Locking Duration' with the value '2' and a 'Day(s)' dropdown. To the right is a checkbox for 'Apply Legal Hold'. Below these are a 'File Name Prefix' field with '(optional)' and a close icon, and a 'Metadata' section with a '+ Add' button and a table containing 'x-lock-meta' and 'Object Locking', each with a close icon. At the bottom is a '+ Add or drop files' button.

3. A legal Hold can also be applied during the upload. This setting is independent of the retention locking settings and is not subject to any bucket-level defaults.
4. Adding a file name prefix and metadata for the object version is optional.
5. Click **Start Upload**.

The new object version is added to the bucket with the retention configuration applied. A lock icon is displayed next to the object version name to represent that the object version is locked.

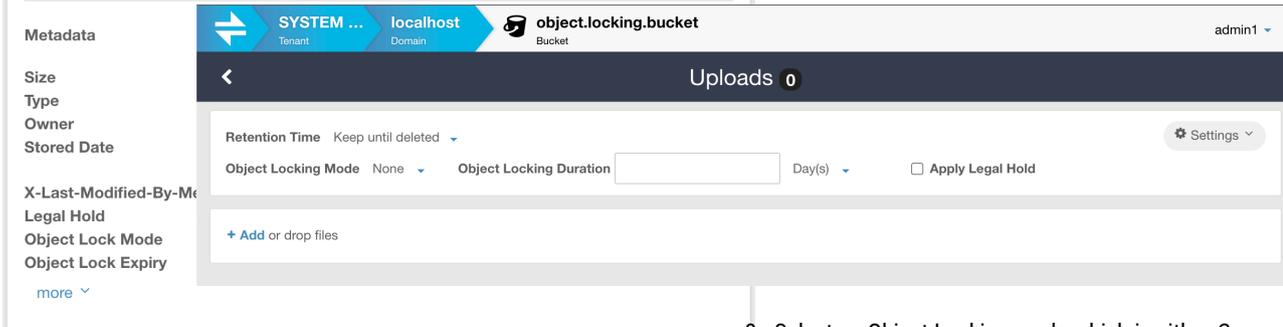


Click on the locked object version to view the Object Locking details.

Setting Up Retention on a New Object

If the bucket has Object Locking enabled but does not have any prior default retention configured, a retention policy can explicitly be set on a new object during its creation. To learn more about uploading through Content UI, see [Uploading Files](#).

1. Navigate to the upload page for the target bucket in Content UI.
2. The upload settings for Object Locking are located under *Retention Time*. As there are no defaults at the bucket level, no policy is automatically applied. An object lock policy is optional; no policy needs to be defined. If any values are specified, then the upload does not proceed until all required values are present.



3. Select an Object Locking mode which is either *Governance* or *Compliance*.

4. Enter the Object Locking duration in either days or years.
5. A legal hold can also be applied during the upload. This setting is independent of the retention locking settings and is not subject to any bucket-level defaults.
6. Adding a file name prefix and metadata for the object version is optional.
7. Click **Start Upload**.

The new object version is added to the bucket with the retention configuration applied. A lock icon is displayed next to the object version name to represent that the object version is locked.

Click on the object version to view its details.

Modifying Retention on an Existing Object Version

The user can apply a retention lock on an object version as long as the bucket has Object Locking enabled if an existing object version does not have Object Locking enabled. This same process is used to update Object Locking on an existing object version. Refer to the following steps to apply retention on an existing object:

1. Locate the object within the bucket.

SYSTEM ... localhost object.locking.bucket
Tenant Domain Bucket

Objects 5

2. Select the object and optionally choose a specific version to update.
3. Click the *Actions* drop-down in the details for the object.
4. Select *Edit Object Lock* from the list.

Storage 55 MB raw | Bandwidth 398 KB past month

Filter Objects

Name

- Basic_drum_pattern_with_128th_note_ride.mid
- Snip20211123_1.png
- api.log

Version 2021-12-15 6:34:49 PM UTC

Metadata

Size	167.94 KB
Type	(none)
Owner	bguetzlaff@
Stored Date	2021-12-15 6:34:49 PM
X-Last-Modified-By-Meta	bguetzlaff@
Legal Hold	On

api.log

Version 2021-12-15 6:38:50 PM UTC

Metadata

Size	167.94 KB
Type	(none)
Owner	bguetzlaff@
Stored Date	2021-12-15 6:38:50 PM
X-Last-Modified-By-Meta	bguetzlaff@

Actions

- Create collection...
- Edit metadata...
- Edit object lock...
- Place under legal hold...

5. Select the retention mode which is either Governance or Compliance.

Retention Locking

Current Lock Status: Unlocked

MODE

Unlocked

Governance

Compliance

Cancel Update Lock

6. Enter a retention duration in days or years.
7. Click **Update Lock**.

Content UI shows a success message saying that ***Object Lock status updated*** once the retention is set up.

Applying Legal Hold

There are two ways to apply a legal hold to an object version:

- Explicitly enable legal hold when creating a new object.
- Explicitly enable legal hold on an existing object version.

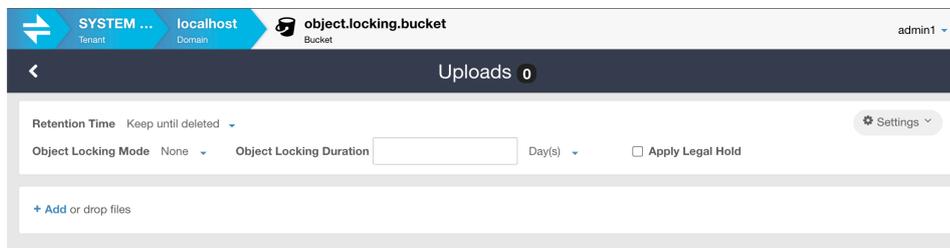
Info

One can apply a legal hold irrespective of retention locks applied on the object version. It is applied independently of retention locks.

Setting Up Legal Hold During Upload

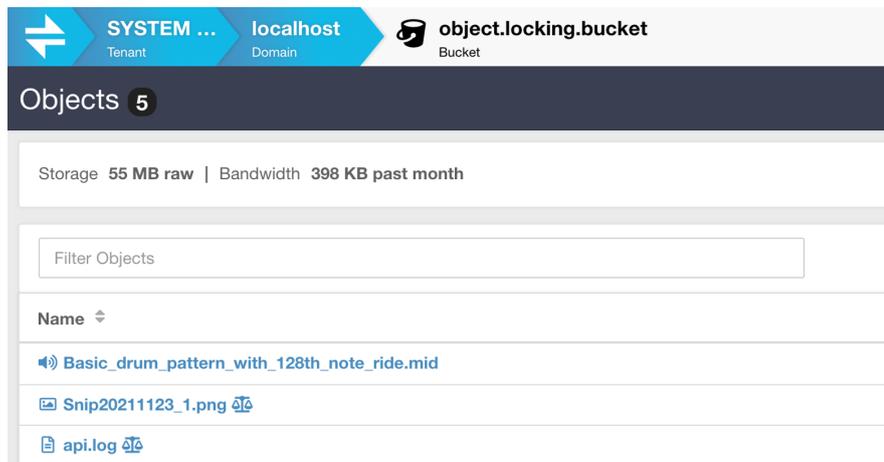
The legal hold can explicitly set on a new object during creation if the bucket has Object Locking enabled. See [Uploading Files](#) to learn more about uploading through Content UI.

1. Navigate to the upload page for the target bucket in Content UI.
2. The upload settings for Object Locking are located under *Retention Time*.



3. Select the checkbox for *Apply Legal Hold*. This setting is independent of retention locking settings.
4. Adding a file name prefix and metadata for the object version is optional.
5. Click **Start Upload**.

The new object version is added to the bucket with the legal hold applied. A “balance scales” icon is displayed next to the object version name to represent that the object version has the legal hold applied.



Click on the object version to view the details.

Modifying Legal Hold on an Existing Object Version

One can apply the legal hold to an existing object version as long as the bucket has Object Locking enabled. This same process is used to remove the legal hold from an existing object version. Refer to the following steps to modify legal hold on an existing object:

1. Locate the object within the bucket.
2. Select the object and optionally choose a specific version to update.

Version

Metadata Actions ▾

Size
Type
Owner
Stored

X-Last-Modified-By-Meta
Legal Hold
Object Lock
Object Lock
[more](#)

3. Click the *Actions* drop-down in the details for the object.

4. Select *Apply Legal Hold* from the list. This reads *Remove Legal Hold* if the legal hold is already applied.

api.log
Share
✕



Version

Metadata Actions ▾

Size	167.94 KB
Type	(none)
Owner	bguetzlaff@
Stored Date	2021-12-15 6
X-Last-Modified-By-Meta	bguetzlaff@

[more](#) ▾

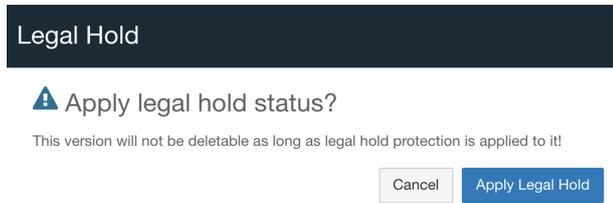
🔍 Create collection...

✎ Edit metadata...

🔒 Edit object lock...

⚖️ Place under legal hold...

5. Verify the change of legal hold status.



6. Click **Apply Legal Hold**. This reads **Remove Legal Hold** when removing legal hold.

Content UI shows a success message saying ***Object legal hold applied*** once the legal hold status has changed.

Swarm Hybrid Cloud

- [Overview](#)
- [Capabilities](#)
- [Prerequisites](#)
- [Usage](#)
- [Workflow](#)
- [Content UI](#)
 - [Replicating data to the destination](#)

Overview

Swarm Hybrid Cloud provides the capability to copy objects from Swarm to a target S3 cloud storage destination. Native cloud services and /or applications running on utility computing can work with data directly from the target cloud storage. Future releases provide additional capabilities.

Capabilities

- The Content UI provides access to the capabilities.
- It functions with the AWS S3 object storage service and supports general S3 buckets as the target cloud storage.
- It uses the provided target bucket with an S3-valid name, endpoint, access key, and secret key.
- It uses a selected dataset to copy to the cloud (known as the 'focus dataset'). This can be a bucket or dynamic set of criteria from a collection of the Swarm source.
- The focus dataset remains in the native format in the target cloud storage, and users/applications/services can read the focus dataset directly in the target cloud storage without any additional steps.
- The integrity of all objects in the focus dataset is preserved as it is copied to the target cloud storage. Status is provided with the result of each copy from the focus dataset for reference and verification.
- The object metadata can be preserved, modified, or removed in part depending on the target storage system.
- Objects are transferred securely.
- The initial release enables the focus dataset transfer to be initiated on a manual basis.

Info

Each object from the focus dataset is copied to the cloud and does not *move* to the cloud. They remain within the Swarm namespace as the authoritative copy and remain searchable in the Swarm namespace after the copy.

- The "folder" path of each object is preserved.
- The payload size and hash integrity are checked at the target cloud storage and reported in the log file.
- The target bucket can exist already or it can be created using the provided credentials. Either way, target credentials need permission for bucket creation. This functionality may change in future releases.

Prerequisites

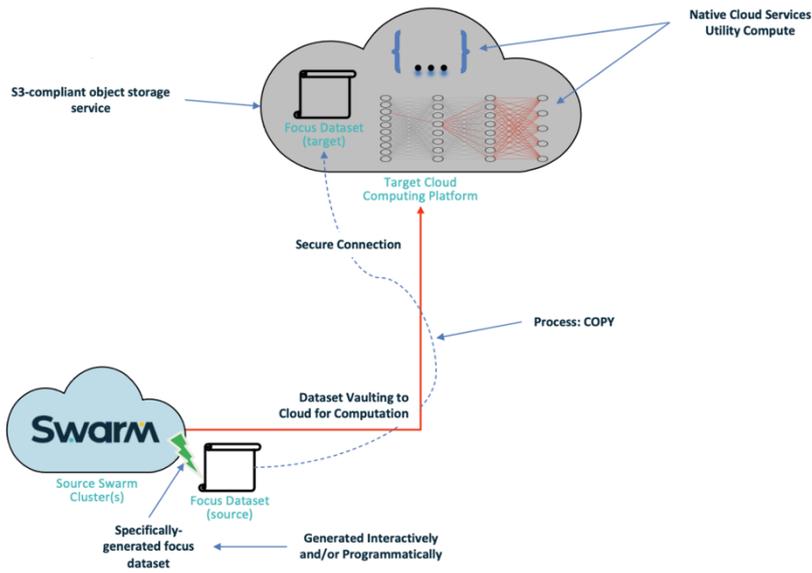
- At least one bucket with focus dataset to copy and is created on Swarm UI
- At least one bucket is created on the target cloud storage service.
- Token ID and S3 secret key generated on the target cloud storage service.

Usage

Clients can access this feature through the Content UI. Clients are required to select a specific dataset to copy to the cloud, which can be either a collection or bucket. Provide the target bucket details, e.g., endpoint, access key & secret key. Results for each object are provided in the source bucket as a status file. The focus dataset is defined shortly after the job is triggered and is not redefined during execution. The job can be reviewed from the generated dictionary and log files.

Workflow

HYBRID CLOUD INITIATIVE



Content UI

Hybrid Cloud helps in replicating the focus dataset, therefore, the client needs two environments, one is Swarm UI and another is target cloud storage service. The client can copy all data from the source location (Swarm) and make a copy of the same data at the destination (client's target cloud storage service). It is applied at the bucket level. The entire data residing in a bucket can be copied and placed in the destination. This whole process of replicating data needs job creation at the source location.

Create a job at the bucket level to start copying the focus dataset to the target cloud storage service.

Irrespective of the focus dataset copied successfully or not, there are two or more files created after the job submission:

- **Manifest File** - Contains information such as total object count & size, endpoint, target bucket name, access key, and secret key of focus dataset copied.
- **Dictionary File(s)** - Contains the list of all focus datasets copied. They show the name of each object along with the size in bytes.
- **Log File** - Provides the current status of each object of the focus dataset copied, along with the details from the final check. This file is generated once the objects are queued up to copy, and refreshed every two minutes. There are four potential statuses for each object:
 - **Pending** - This is the initial state. The object is queued up for the copy.
 - **Failed** - The object failed to copy to the target cloud storage service. Reasons include the target endpoint is not accessible, or there can be a problem with an individual object, such as too large an object name for S3. See the Gateway server log for failure details.
 - **Copied** - The object successfully copied.
 - **Skipped** - The object was skipped. Reasons include the object already exists at the destination, the object did not exist at the source, or the object was marked for deletion and shouldn't be copied.

Each file has an importance and provides information about the focus dataset copied from the Swarm UI to the target cloud storage service. The format of these may change in future releases. The Manifest and Log files are overwritten if the same job name is used from a previous run, so save off the files or use a different job name if this is not desired.

After the copying has started, any or all files can be renamed if needed. Any log file updates continue under the old name.

Replicating data to the destination

Refer to the following steps to replicate the focus dataset:

1. Navigate to the Swarm UI bucket or collection to copy.
2. Click Actions (three gear icons) and select *Copy to S3*. A modal presents a form, with required fields marked with asterisks (*). Here is an example:

Storage Date	Size	Type
2022-04-13 6:11:49 PM	187.76 MB	application/zip
2022-04-13 6:12:04 PM	244.74 MB	application/zip
2022-04-13 6:21:30 PM	507.84 MB	application/zip
2022-04-13 6:21:36 PM	1.12 GB	application/zip
2022-04-13 6:07:26 PM	1.20 MB	application/zip

⚙️ Actions 🔍 Search 🔄 Refresh

Copy to S3...

Copy to S3

Job Name *

Endpoint *

Region

Bucket *

Access Key *

Secret Key *

- Job Name - Provide a unique name for the job. The Manifest and Log files are overwritten if a job name is reused from a previous run.
 - End Point - This is the target service endpoint.
 - For AWS S3 targets - The format is shown in the screenshot above.
 - For Swarm targets - The value needs to be in the following format with HTTP or HTTPS as needed:
`http://${S3_PROTOCOL}://${DOMAIN}:${S3_PORT}`
 - Region - The S3 region to use. Some S3 providers may not require this.
 - Bucket - Enter the target bucket name.
 - Access Key - This is the access key for the target bucket. This needs to be generated within the target cloud storage service.
 - Secret Key - This is the S3 secret key and is generated with the access key.
3. Click **Begin S3 Copy**. This button is enabled once all required text fields are filled.

The copy operation generates the earlier objects (manifest, dictionary object, and log). All use the given job name as a prefix but are appended with separate suffixes. The duration of the job directly depends on the size of the job (the count of objects and the total number of bytes to be transferred). To monitor the status of the job, download and open the latest copy of the status log.

Bucket Lifecycle Policy

- [S3 Lifecycle Policy Features at Bucket Level](#)
 - [Use Cases](#)
- [Swarm Lifecycle Policy](#)

Lifecycle policies perform active deletion of the non-locked content based on time-based policies. Though object locking and lifecycle policies are not direct mirrors of each other, having both capabilities allows a customer to prevent deletion-style ransomware attacks while providing more complex protection schemes.

S3 Lifecycle Policy Features at Bucket Level

- A bucket contains a set of overlapping policies called rules.
- Each rule contains an ID (name).
- Each rule contains a flag indicating if it is enabled or not.
- Each rule contains an optional filter describing the rule applied to which objects within the bucket.
 - The absence of a filter indicates all objects in the bucket are subject to the rule.
 - A filter includes a prefix of an object name (relative to the bucket) that is subject to the rule.

 AWS S3 supports tag-based filters, but Swarm does not support them.

- A rule contains an expiration action indicating the deletion of an object when the rule applies. The expiration action operates on the current version of an object, either by deleting the object permanently in a non-versioned bucket or creating a delete marker in a versioned bucket.
 - A time limit is provided in the form of *“integral number of days”* or *“absolute date”*.
 - If an absolute date is provided, any object created before that date is subject to expiration.
 - When an integral number of days is provided, the evaluation of the [expiration cut-off](#) for the current version uses the object version’s age, rounding up to midnight UTC. The integral number of days starts from 0 at that time. For historic (non-current) object versions, the date of the next newest object version is used for evaluation, again rounded up to midnight UTC.
 - In both cases, the policy takes effect at midnight UTC of the specified date.
 - Policies based on the number of object versions are not yet supported.
- Swarm supports `policy.lifecycle` policy setting at the cluster level that acts as a master switch for the feature.

Use Cases

- S3 stores various lifecycle policies at the granularity of buckets with policies applying to the content in those buckets. Since policies apply to all objects in a bucket, Swarm provides an ability to apply the same S3 behaviors to the content stored and accessed via other protocols (e.g., SCSP).
- Object versioning provides durability, but Swarm previously lacked a convenient mechanism to prevent old versions from unbounded accumulation. Lifecycle policies allow a bucket owner to limit version accumulation and thus, mitigate concerns about runaway cluster space usage.

Swarm Lifecycle Policy

Swarm allows all client applications to protect data from malicious deletions and overwrites, without any middle layer. The Swarm UI provides the simplest way to enable or disable lifecycle policies for buckets within the domain.

Swarm is constantly checking and reviewing Bucket lifecycle policies to verify the policies are enforced in a timely manner. See [Supported Amazon S3 Features](#).

Recommended

- To specify multiple rules, use multiple Policy-Lifecycle headers on the bucket, one for each rule. If needed, place multiple rules into a single Policy-Lifecycle header, with each rule separated by commas.

- [Design & Technical Specifications](#)
- [Lifecycle Policy Usage & Examples](#)
- [Client Interfaces](#)
- [Install & Uninstall Instructions](#)

Design & Technical Specifications

Bucket lifecycle policies are designed to offer the following technical specifications:

- [Policy Specification](#)
- [Evaluation](#)
- [Enforcement](#)

Policy Specification

- [Cluster Setting Values](#)
- [Domain Setting Values](#)
- [Bucket Setting Values](#)
- [Supported Rule Attributes](#)
- [Rules with Attributes](#)
 - [Expiration Time Rule](#)

Lifecycle policy specification includes:

- Cluster setting
- Policy header on domain objects (optional)
- Policy headers on bucket objects

Cluster Setting Values

The Swarm cluster setting `policy.lifecycle` supports two values:

- **disabled** – A default value for the evaluation of all lifecycle policies in the cluster to provide legacy behavior.
- **enabled** – Either enable or disable lifecycle policies at the domain level for domains where such policies are applied.

Use *Management API* for `policy.lifecycle` setting.

Domain Setting Values

Domain objects support a `Policy-Lifecycle` header to control the behavior of lifecycle policies for buckets in the domain. The header supports either of the following values:

- **<unspecified>** – The lack of a defined policy header represents that lifecycle policies are enabled for buckets in the domain when the `policy.lifecycle` setting is enabled. Applying lifecycle policy at the domain level is optional.
- **enabled** – The lifecycle policies are enabled for buckets in a domain when the `policy.lifecycle` cluster setting is enabled.
- **disabled** – The lifecycle policies are disabled for buckets in a domain regardless of `policy.lifecycle` setting.

 Lifecycle policy is not applied to unnamed content within a domain. Only named objects within buckets contains lifecycle policy applied.

Bucket Setting Values

Bucket objects support a `Policy-Lifecycle` header with multiple values.

- Each header value encodes one lifecycle policy rule.
- Each lifecycle rule is comprised of a number of optional attributes, expressed as `<name>:<value>` pairs separated by space. Extra spaces are allowed at the beginning, end, and before & after the colon.

Important

- Duplicate names are not allowed across lifecycle rules for a bucket.

- Unsupported names or values return an HTTP 400 error on the bucket (or domain) write. The 400 response indicates the source of the problem.

Supported Rule Attributes

Attribute	Value	Definition
RuleId	<unique rule id>	A required, user-defined ID of the rule. The value is contained within quotes and must be URL-encoded.
Enabled	<true false>	An optional indication to verify if the rule is enabled. The absence of this attribute indicates the rule enabled.
NamePrefix	<prefix>	An optional prefix to match against the relative name of the object in question. <ul style="list-style-type: none"> • Always use quoted value and verify it is URL encoded • Never use slash as a first character for the prefix. • The rule is applied to the object if the prefix is matched with the object name. • The absence of the prefix indicates the rule is applied to all objects in the bucket.
ExpirationDays	<nonnegative integer>	The current version of an object is expired after the defined number of days.
ExpirationDate	<ISO 8601 date>	The current version of an object is expired after the defined date (midnight UTC time).
ObsoleteExpirationDays	<nonnegative integer>	A non-current version of an object is expired after the defined number of days when the object becomes non-current. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> i This rule takes effect if versioning is enabled on the bucket. </div>
ObsoleteExpirationDate	<ISO 8601 date>	A non-current version of the object is expired after the defined date (midnight UTC time). <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> i This rule takes effect if versioning is enabled on the bucket. </div>

Rules with Attributes

Every rule:

- Must have one or multiple expiration attributes.
- *ExpirationDays* and *ExpirationDate* attributes are mutually exclusive.
- *ObsoleteExpirationDays* and *ObsoleteExpirationDate* attributes are mutually exclusive.

Expiration Time Rule

- For expiration days,
Expiration time = Creation time of the current version + Number of days indicated + Rounded up to the next midnight UTC
- For obsolete expiration days,
Expiration time = Create time of the next newest object version + Number of days indicated + Rounded up to the next midnight UTC

- ISO 8601 dates must unambiguously specify a calendar date. The (unspecified) expiration time is always midnight UTC of that date; any timezone specification is not allowed.
- Expiration of a current version of an object (i.e. non-delete marker in the versioning enabled bucket) represents creating a delete marker, pushing the current version down the versioning stack.
- In all other cases, the object or object version is permanently deleted.

The gateway supports SCSP reads & writes of domain and bucket headers with lifecycle policies specified. Gateway S3 interface is modified to support GET, PUT, DELETE, and [related permissions](#) (GetLifecycleConfiguration and PutLifecycleConfiguration) for bucket lifecycle policies as specified in the [S3 documentation](#). Gateway validates policies against the S3 specification. On PUT or DELETE permission, the gateway translates the client-supplied bucket policy specifications into the appropriate Swarm bucket headers. Bucket lifecycle policy features provided using S3 not supported by Swarm (such as storage class transitions) are dropped during this translation. On the bucket lifecycle policy GET reply, Gateway performs reverse translation for any `Policy-Lifecycle` headers on the bucket object into an S3-compatible format.

Swarm Content Portal provides a convenient method of managing policies. This information is provided for completeness.

i Since lifecycle policies are part of the overall context-level policy framework, GET and HEAD requests on contexts and name objects return `Policy-Lifecycle-Evaluated` and `Policy-Lifecycle-Evaluated-Constrained` headers. These headers describe if the lifecycle policies are enforced at the various levels such as cluster, domain, and bucket.

Evaluation

- [Policy Precedence](#)
- [Policy Evaluation](#)
 - [Evaluation Results for Objects within a Bucket](#)

Policy Precedence

- When the policy is disabled at the cluster level, the cluster-level lifecycle policy takes higher precedence over the domain and bucket-level policy specifications.
- Domain-level lifecycle policies take precedence over bucket-level lifecycle policies.
- Object-level `Lifepoint` headers (indicating non-deletion) take precedence over bucket-level lifecycle policies.

ⓘ Object locking is expressed using object-level `Lifepoint` header(s). This enables object locking to have higher precedence than lifecycle policies.

- The first lifecycle policy that indicates expiration or any object-level `Lifepoint` headers (indicating deletion) determines the actual expiration date.

Policy Evaluation

Bucket lifecycle policies leverage an existing context-level policy evaluation mechanism, but with some caveats.

Evaluation Results for Objects within a Bucket

policy.lifecycle cluster setting	Policy-Lifecycle domain header	Policy-Lifecycle bucket header	Policy enabled/disabled
disabled	<any>	<any>	disabled
enabled	enabled	<present>	enabled
enabled	<unspecified>	<present>	enabled

Important

- Keep the `policy.lifecycle` cluster setting and bucket lifecycle policies always activated to apply to an object within the bucket.
- The `Policy-Lifecycle-Evaluated` response header value is either "enabled" or "disabled", but the actual policies are applied live on the bucket `Policy-Lifecycle` header.

The `Lifepoint` headers take precedence over lifecycle policies on an object. An object that inhibits deletion using `Lifepoint` headers is considered live, but the lifecycle policy header evaluation is not performed. The policy evaluation framework determines whether the lifecycle policies are in effect for the object or not, based on the above table. The lifecycle policies are parsed and applied to the object if so. Within each expiration type (current or past version), the earliest expiration date is applied using any policy rule. Swarm tolerates unknown `name:value` attributes on lifecycle policies discovered on domains and buckets in the cluster to support future-proofing and rolling upgrades.

Enforcement

- [Policy Enforcement](#)
- [Policy Action](#)

Policy Enforcement

- Consider lifecycle policies for an object during SCSP operations. The expiration policies are evaluated to verify the object is “live” before proceeding with the SCSP operation.
- In a versioning-enabled bucket, SCSP operations on objects apply an expiration rule to trigger an action i.e. create delete markers before the intended behavior of the operation.
- Continuously check objects in the background to verify compliance with lifecycle policies.
- Listing queries are not required to evaluate rules for objects returned in the listing. Therefore, listing queries return inconsistent results immediately after policy expiration at midnight UTC when compared with the active SCSP operations.
- Listing queries provide an eventual consistency subject for background policy evaluation.

Policy Action

The action is the deletion of an object or object version. In the scenario of a current object deletion, deletion represents creating a delete marker when the bucket/domain has versioning enabled. The delete marker contains a timestamp consistent with the lifecycle policy expiration time, perhaps the creation of the delete marker occurred.

A liveness check is performed on the object as part of SCSP requests for `Lifepoint` headers (policies).

The policies are set up to “fire” at midnight UTC; this requires components and processes provided by gateway installation. Listing requests for expired objects appear out-of-date in some cases but are eventually consistent over time.

Lifecycle Policy Usage & Examples

A user must enable lifecycle policies at the cluster setting level, a. When enabled, use the management API:

```
curl -X PUT --header 'Content-Type: application/json' --header 'Accept: text/plain' -d '{}'  
'http://lucky1.tx.caringo.com:91/api/storage/clusters/_self/settings/policy.lifecycle?value=enabl
```

 A missing `Policy-Lifecycle` header on the domain is considered a tacit enablement which does not require any changes.

A bucket contains one or multiple lifecycle policies, such as:

- Expire all versioned content after one year
- Expire all current content after 5 years, etc.

Such policies are applied on a bucket object using one or multiple `Policy-Lifecycle` headers:

```
curl -X COPY --post301 --location-trusted -H 'Policy-Lifecycle: RuleId:"rule5" ExpirationDays:182'  
'http://lucky1.tx.caringo.com/mybucket?domain=mydomain&preserve'
```

In the previous example, apply implicitly to all content in the bucket by;

- Naming the single policy
- Not declaring it enabled (default enabled)
- Relying on a missing prefix

The preserve query argument on the COPY operations indicates leaving other persisted headers with no change.

Important

- Re-transmit all `Policy-Lifecycle` headers to appear on the new object. It is advised to use Content UI for editing policy rules.
- Currently, Swarm supports expiration policies.

See [S3 lifecycle policy examples](#).

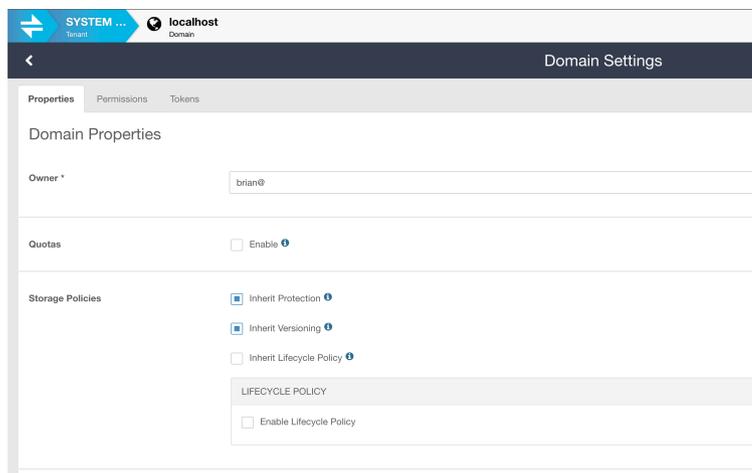
Client Interfaces

The client interface in Swarm consists of:

- `policy.lifecycle` [setting at the cluster-level](#)
- `Policy-Lifecycle` [header on a domain](#)
- `Policy-Lifecycle` [headers on a bucket](#)

The `Policy-Lifecycle-Evaluated` headers return the evaluated policies on domains, buckets, and objects using GET/HEAD requests. For named objects, the verbose query argument is required to view those headers. The gateway supports S3 lifecycle policies as described in the [S3 documentation](#).

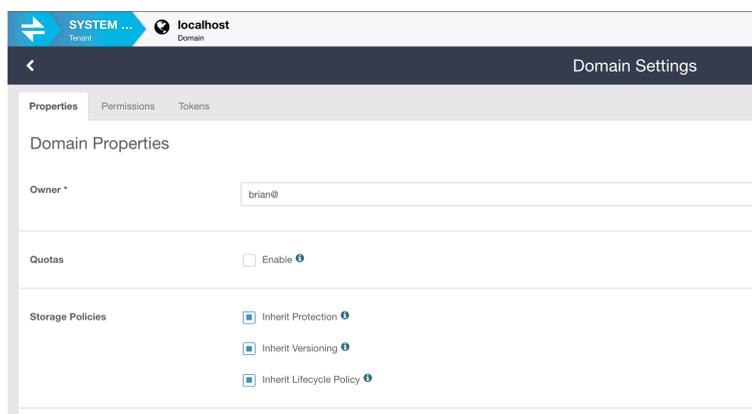
Swarm UI supports the above cluster setting just like any existing cluster setting. On the Content Portal, a customer can edit at the domain and bucket levels. Further, a customer can enable/disable lifecycle policies for buckets within the domain using a checkbox.



The default setting for a domain is to inherit lifecycle policies from the cluster.

The default cluster-level setting is disabled.

Time-based policies trigger the defined rule(s) when the defined duration completes.



SYSTEM ... localhost lots.of.stuff

Bucket Settings

Properties Permissions

Bucket Properties

Owner *

Quotas Enable ⓘ

Storage Policies Inherit Protection ⓘ
 Inherit Versioning ⓘ
 Enable Object Locking ⓘ

Lifecycle Policy **Override Alert:** Lifecycle policy evaluation in this domain is currently disabled.
Policy Rules ⓘ
[+ Add rule](#)

Lifecycle Policy

Policy Rules ⓘ

RULE: CLEAN UP AFTER 90 DAYS

Name * Rule enabled

Object Name Prefix

ACTIONS

Current Version Expiry None After days After date

Expiry Days *

Historical Versions Expiry None After days After date

[+ Add rule](#)

SYSTEM ... localhost lots.of.stuff

Bucket Settings

Properties Permissions

Bucket Properties

Owner *

Quotas Enable ⓘ

Storage Policies Inherit Protection ⓘ
 Inherit Versioning ⓘ
 Enable Object Locking ⓘ

Lifecycle Policy Policy Rules ⓘ
[+ Add rule](#)

Install & Uninstall Instructions

To support the bucket lifecycle feature, use:

Components	Version
Gateway	7.9
Content Portal	7.6
Swarm Storage	14.1.0

Documentation Archive

© 2005–2022 [DataCore Software Corporation](#). All rights reserved.

[Open Source Software Licenses](#) | [EULA](#) | [Support](#)

No part of this material may be reproduced, transmitted, or transcribed without the written consent of DataCore Software Corporation, Inc.

Updates may be made to this material and incorporated into later editions.

- **Swarm 14** | [Select the PDF by bundle date, below](#)
- [Swarm Deployment](#)
- [FileFly 4](#) | [Release Notes](#), [Administration Guide](#)
- Swarm 12, 11, Swarm 10, Swarm 9 | [Select the PDF by bundle date, below](#)
- SCSP Proxy | [Release Notes](#), [Overview](#), [OSS Licenses](#)
- [CloudScaler](#) | [Release Notes](#), [Guide](#)

The online documentation always reflects the newest released version; use the PDF links below for specific software bundle releases.



Bundle date

The PDF may be corrected after the bundle is released, but the date in the PDF's *filename* does not change: it is there to point to the Swarm software bundle to which it applies.

- [Bundle date](#)
 - [Swarm 14](#)
 - [Swarm 12](#)
 - [Swarm 11](#)
 - [Swarm 10](#)
 - [Swarm 9](#)

Swarm 14	Swarm Storage	SCS	ES, Metrics, Search	Gateway	Content UI	Storage UI	SwarmFS
2022/04/20 Docs PDF	14.1	SCS 1.2	ES: 7.5.2 Search: 7.0.4 Metrics has been deprecated.	7.9	7.6	3.4	3.2
2022/02/04 Docs PDF	14.0.1	SCS 1.1	ES: 7.5.2 Search: 7.0.3 Metrics has been deprecated.	7.8	7.5	3.4	3.2
2021/12/16 Docs PDF	14.0.1	SCS 1.0	ES: 7.5.2 Search: 7.0.3 Metrics has been deprecated.	7.7	7.4	3.4	3.2

2021/10/15 Docs PDF	14.0.1	SCS 1.0	ES: 7.5.2 Search: 7.0.2 Metrics has been deprecated.	7.6 Object locking Improved S3 listings Multi-delete	7.4	3.3	3.2
Swarm 12	Swarm Storage	CSN	ES, Metrics, Search	Gateway	Content UI	Storage UI	SwarmFS
2021/05/20 Docs PDF	12.1.0	8.3.2	7.5.2, 7.0.1, 7.0.1	7.5 Settings updates, improved error handling	7.4	3.3	3.2
2021/05/12 Docs PDF	12.1.0 Defragmentation optimizations, enhancements to overlay index initialization, operational improvements, 3rd party software updates	8.3.2	7.5.2, 7.0.1 , 7.0.1	7.4 Configurable Prometheus/Node exporter support	7.4 Drag-and-drop functionality in folder listing views. improved data protection policies control	3.3 Configuration and safety guidance when deleting a primary search feed, chassis serial number in chassis details page	3.2
2021/03/23 Docs PDF	12.0.1	8.3.2	7.5.2, 7.0.0, 7.0.1	7.3 System domain	7.3 System domain	3.2	3.1
2021/02/23 Docs PDF	12.0.1 Memory, range reads fixes	8.3.2	7.5.2, 7.0.0, 7.0.0	7.2	7.2	3.2	3.1
2021/01/31 Docs PDF	12.0.0	8.3.2	7.5.2, 7.0.0, 7.0.0	7.2 SAML for tenants, fixes	7.2 SAML, token fixes	3.2 SAML, token fixes	3.1
2020/12/23 Docs PDF	12.0.0	8.3.2	7.5.2, 7.0.0, 7.0.0	7.1	7.1 RSW, logon fixes	3.1 Feeds, logon fixes	3.1
2020/12/04 Docs PDF	12.0.0 enableVolumeRedirects, S3 Backup to Glacier	8.3.2	7.5.2, 7.0.0, 7.0.0 In-place upgrade from ES 6.8.6	7.1 enableVolumeRedirects, SAML, Password encryption	7.0 Remote sync write, Virtual folders, SSO	3.0 NFS config, SSO	3.1 Virtual folders

Swarm 11	Swarm Storage	CSN	ES, Metrics, Search	Gateway, Service Proxy	Content UI	Storage UI	SwarmFS
2020/08/24 Docs PDF	11.3.0 Hardening	8.3.2	6.8.6, 6.3.1, 6.3.1	7.0 , 6.1 Folder listings	6.3	2.4 NFS config	3.1 Folder listings
2020/06/26 Docs PDF	11.2.0 Next-gen SEND	8.3.2	6.8.6, 6.3.1 , 6.3.1	6.4 , 6.1 Support for SEND	6.3	2.3	2.4
2020/05/08 Docs PDF	11.1.0	8.3.2	6.8.6, 6.3.0, 6.3.0 5.6.12, 5.0.8, 5.0.10	6.3.1 , 6.1 Several fixes	6.3.1 Video clip fix	2.3	2.4
2020/04/23 Docs PDF	11.1.0 ES 6, Python3 New kernel, modernization	8.3.2	6.8.6, 6.3.0, 6.3.0 5.6.12, 5.0.8, 5.0.10 ES 6, maintaining ES 5	6.3 , 6.1 ES 6 support Unentanted objects	6.3 Faster uploads, fixed Add Collection	2.3	2.4 Swarm 11.1, ES 6 support
2019/11/19 Docs PDF	11.0.3 Kernel config fix, ATA disks	8.3.2	5.6.12, 5.0.8, 5.0.10 2.3.3, 2.5-2, 2.5-2	6.2, 6.1	6.2	2.3	2.3

2019/11/11 Docs PDF	11.0.2 Retire rate/duration; HP fixes	8.3.2	5.6.12, 5.0.8, 5.0.10 2.3.3, 2.5-2, 2.5-2	6.2, 6.1 SCSP Proxy replaced	6.2	2.3	2.3
2019/10/18 Docs PDF	11.0.1 Large cluster performance	8.3.2	5.6.12, 5.0.8, 5.0.10 2.3.3, 2.5-2, 2.5-2	6.2, 6.1	6.2	2.3	2.3
2019/10/04 Docs PDF	11.0.0 S3 Backup Feeds	8.3.2	5.6.12, 5.0.8, 5.0.10 2.3.3, 2.5-2, 2.5-2	6.2, 6.1 Node pool management	6.2 Video clipping, Share	2.3 S3 Backup, NFS settings	2.3 Tuning, performance

Swarm 10	Swarm Storage	Platform, CSN	ES, Metrics, Search	Gateway, Service Proxy	Content UI	Storage UI
2019/04/17 Docs PDF	10.2.1 Faster retires, range reads	10.1 , 8.3.2	5.6.12, 5.0.8 , 5.0.10 2.3.3, 2.5-2, 2.5-2	6.1.1, 6.1 Docker 2.7 support	6.1 Chart improvements	2.2 Access to advanced settings
2019/02/11 Docs PDF	10.1 Faster writes and EC reads Prometheus preview	10.0 , 8.3.2 Upgrade from CSN	5.6.12, 5.0.7, 5.0.9 2.3.3, 2.5-2, 2.5-2	6.0, 6.0 Requires ES 5.6	6.0	2.1 Fixes to login, global settings
2018/12/21 Docs PDF	10.0 New architecture	9.1, 8.3.2 Rolling reboots UI	5.6.12 , 5.0.7 , 5.0.7 2.3.3, 2.5, 2.5	5.4 , 5.3.0 Metadata translation	6.0 UX improvements	2.0 SSL replication, hardware tools

Swarm 9	Swarm Storage	Platform, CSN	ES, Metrics, Search	Gateway, Service Proxy	Content UI	Storage UI
2019/11/26 Docs PDF	9.6.4	9.1, 8.3.2	2.3.3, 2.5, 2.5	5.4.1 , 5.3.0 Support for Swarm UI 2.3, S3 multipart upload fixes	6.0	2.3 S3 Backup; NFS tuning <i>S3 backup requires Swarm 11</i>
2019/02/27 Docs PDF	9.6.4	9.1, 8.3.2	2.3.3, 2.5, 2.5	5.4 , 5.3.0 Metadata translation	6.0 UX improvements	2.1 Fixes to login, global settings, SSL replication, hardware tools
2018/12/12 Docs PDF	9.6.4 Intel driver updates: i40e, ixgbe	9.1, 8.3.2	2.3.3, 2.5, 2.5	5.3.3, 5.3.0	5.5.0	1.2.4
2018/08/31 Docs PDF	9.6.3	9.1, 8.3.2	2.3.3, 2.5, 2.5	5.3.3 , 5.3.0 S3 multipart upload/listing fixes	5.5.0	1.2.4
2018/08/29 Docs PDF	9.6.3 Support HPE ProLiant (smartpqi)	9.1, 8.3.2	2.3.3, 2.5, 2.5	5.3.2, 5.3.0	5.5.0	1.2.4

2018/08 /21 Docs PDF	9.6.2 Fixed EC manifest fails to replicate	9.1, 8.3.2 Security improvement	2.3.3, 2.5, 2.5	5.3.2, 5.3.0	5.5.0	1.2.4
2018/07 /30 Docs PDF	9.6.0	9.1, 8.3 RHEL/CentOS 6.10	2.3.3, 2.5, 2.5	5.3.2 , 5.3.0 Bug fixes	5.5.0	1.2.4
2018/06 /29 Docs PDF	9.6.0 New replication method, OSS updates	9.1 , 8.3 UEFI support, rolling reboots, new CLI	2.3.3, 2.5, 2.5	5.3.1 , 5.3.0 New replication method	5.5.0 Object renaming, Delete current version	1.2.4 New replication method, SwarmFS 2 support
2018/05 /15 Docs PDF	9.5.3 Optimized retries, Header filtering, OSS updates	9.0, 8.3	2.3.3, 2.5 , 2.5 Improved indices	5.2.5, 5.2.3-2	5.4.0 Metadata editor, Delete versioned	1.2.3 SwarmFS export config
2018/01 /12 Docs PDF	9.4.0 Elasticsearch config script, OSS updates (jessie)	9.0, 8.3	2.3.3, 2.4, 2.4-3	5.2.5 , 5.2.3-2 Enhanced listing consistency, S3 compatibility support	5.3.1 Access policy editor	1.2.1 SwarmFS 1 support

Swarm Release Notes

- [Swarm 14 Highlights](#)
- [Content Gateway Release Notes](#)
- [Content UI Release Notes](#)
- [SDK Release Notes](#)
- [SwarmFS Release Notes](#)
- [Swarm Storage Release Notes](#)
- [Storage UI Release Notes](#)
- [Swarm Platform Release Notes](#)

For information about Platform Server 10, contact [DataCore Support](#).

Swarm 14 Highlights

Swarm combines the scalable software-defined object storage of Swarm Storage with the components to support diverse implementations:

- *Platform Server* – Node for site-wide management and services
- *Storage Cluster* – Cluster for Swarm storage nodes
- *Elasticsearch* – Cluster for search and historical metrics
- *Content Gateway* – Gateway for cloud-based client access (S3)
- *Storage UI* – Website for storage cluster management
- *Content UI* – Website for cloud content management
- *SwarmFS* – Optional connector for NFS clients

- [Swarm 14.1 – updated April 2022](#)
- [Swarm 14 – updated October 2021](#)

Swarm 14.1 – updated April 2022

Storage	Platform/CSN	ES, Search	Gateway	Content UI	Storage UI	SwarmFS
14.1	14.1	7.5.2, 7.0.2	7.9	7.6	3.3	3.2

- **Improved data protection through new capability in the content lifecycle policy.**
- **Ease of deployment with Swarm Cluster Services (SCS) 2.0.**

Swarm 14 – updated October 2021

Storage	Platform/CSN	ES, Search	Gateway	Content UI	Storage UI	SwarmFS
14.0.1	14.0	7.5.2, 7.0.2	7.6	7.4	3.3	3.2

- **Simplified cluster configuration and storage node deployment in newly released Platform 14.**
- **Object locking & Improved S3 listings in Gateway 7.6**
- **Erasure Coding Improvements in Swarm Storage 14.0**

- [Swarm 12 Highlights](#)
- [Swarm 11 Highlights](#)

Swarm 12 Highlights

i Swarm combines the scalable software-defined object storage of Swarm Storage with the components to support diverse implementations:

- *Platform Server* – Node for site-wide management and services
- *Storage Cluster* – Cluster for Swarm storage nodes
- *Elasticsearch* – Cluster for search and historical metrics
- *Content Gateway* – Gateway for cloud-based client access (S3)
- *Storage UI* – Website for storage cluster management
- *Content UI* – Website for cloud content management
- *SwarmFS* – Optional connector for NFS clients

- [Swarm 12.1 – updated May 2021](#)
- [Swarm 12.1 – launched May 2021](#)
- [Swarm 12.0 – updated March 2021](#)
- [Swarm 12.0 – updated February 2021](#)
- [Swarm 12.0 – launched December 2020](#)

Swarm 12.1 – updated May 2021

Storage	CSN Platform	ES, Metrics, Search	Gateway	Content UI	Storage UI	SwarmFS
12.1.0	8.3.2	7.5.2, 7.0.1, 7.0.1	7.5	7.4	3.3	3.2

- **Settings updates in Content Gateway 7.5**
- **Improved error handling in Content Gateway 7.5**

Swarm 12.1 – launched May 2021

Storage	CSN Platform	ES, Metrics, Search	Gateway	Content UI	Storage UI	SwarmFS
12.1.0	8.3.2	7.5.2, 7.0.1, 7.0.1	7.4	7.4	3.3	3.2

- **Defragmentation optimizations** – These optimizations include additional parallelism across disk volumes for nodes with many drives and new tuning features for small-object/high-turnover use cases.
- **Enhancements to overlay index initialization** – These improve the startup performance at reboot by allowing a node's index to more quickly reach an authoritative state and reduce the disruption of rolling cluster reboots.
- **Operational improvements** – This release contains several optimizations for administrative operations, better statistical reporting, and expanded diagnostic information.
- **3rd party software updates** – Third party software packages within Swarm are updated to track with CVE recommendations.

Swarm 12.0 – updated March 2021

Storage	CSN Platform	ES, Metrics, Search	Gateway	Content UI	Storage UI	SwarmFS
12.0.1	8.3.2	7.5.2, 7.0.0, 7.0.1	7.3	7.3	3.2	3.1

- **System Domain** – In the Content UI, Swarm's modern features such as metadata searching, policy/access control, and metering for unnamed and untenanted objects can now be used. See [System Domain and Legacy Mode for Gateway](#) for more on this feature.
- **Elasticsearch** – Improvements to the initialization of the metadata search indices to verify Elasticsearch's automatic field data typing is correct.

Swarm 12.0 – updated February 2021

Storage	CSN Platform	ES, Metrics, Search	Gateway	Content UI	Storage UI	SwarmFS
12.0.1	8.3.2	7.5.2, 7.0.0, 7.0.0	7.2	7.2	3.0	3.1

- **Single sign-on enhancements** – The SAML single sign-on support in Gateway and Content UI is enhanced to allow better mixing of identity management systems for separate tenants in Swarm.
- **Swarm Improvements** – Swarm Storage improvements for 12.0 or upgrades from an earlier version.

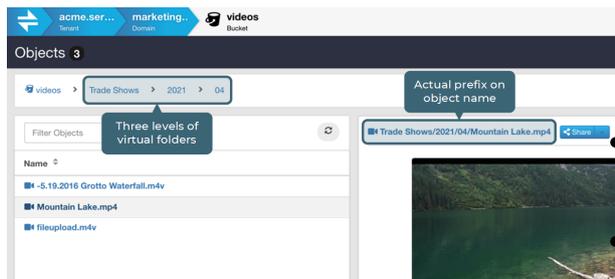
Swarm 12.0 – launched December 2020

Storage	CSN Platform	ES, Metrics, Search	Gateway	Content UI	Storage UI	SwarmFS
12.0	8.3.2	7.5.2, 7.0.0, 7.0.0	7.1	7.0	3.0	3.1

- **Remote Synchronous Write** – In the Content UI, specific domains and buckets can now be configured to broadcast new content to all remote sites immediately. This feature, *Remote Synchronous Write*, delays write completion until replicas exist in every remote cluster. This setting allows supporting applications that require guarantees that backups are committed to every site, and to support publishing requirements to be able to read new content from any remote site immediately after ingest.



- **Folder Listing UI** – With Gateway 7, folder listing support across Swarm clients (Content UI, SwarmFS, and S3) is rearchitected and centralized. *Folder listing* is what renders the virtual folders (prefixes) on named Swarm objects (such as `FY2019/Q3/object.jpg`) into familiar folders on the users' file systems. The service leverages newer Elasticsearch features and is not bound by ES listing limits.



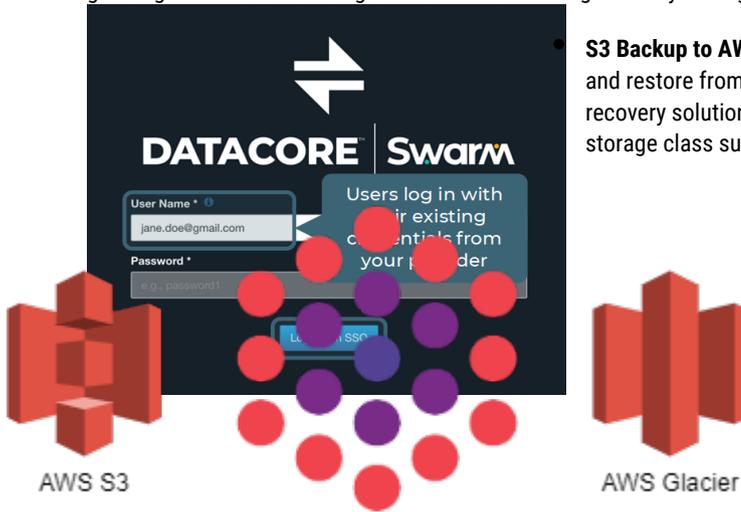
The Content UI now presents these folders as walkable directories, with these key benefits:

- **Prefix filtering** – By parsing object prefixes into hierarchical folders in real time, the Content UI provides users a fast and intuitive way to view and manage content in a bucket, automatically.

- **Empty folders** – The Content UI allows creation and persistence of new, empty folders that are ready to receive files. This allows planning and setting up organizing structures ahead of time, to guide content

uploaders to use the organization. A content architecture can be enforced to avoid the risk users perform bulk uploads using a malformed prefix by having them upload directly to folders.

- **Recursive deletes** – The Content UI allows deleting virtual folders, which recursively deletes all objects *and subfolders* they contain.
- **Single sign-on across UIs** – SAML 2.0 support in Content Gateway facilitates implementing single sign-on for users. Adding SSO allows granting access to the Storage and Content UIs using centrally managed identity credentials, such as the organization's Gmail accounts.



- **S3 Backup to AWS Glacier** – S3 Backup feeds can now be implemented to back up to and restore from archival ("cold") storage. The cost-effectiveness of the disaster recovery solution can be maximized by backing up Swarm to an S3 bucket with a storage class such as Glacier.

Swarm S3 Backup and Restore

- **Elasticsearch 7** – Staying current with the latest version of Elasticsearch supported by Swarm reduces the complexity of cluster upgrades, tooling, and troubleshooting. The [last several Elasticsearch releases](#) brought improvements in performance, resilience, and cost-efficiency, and upgrading to Elasticsearch 7 from 6 does not require cluster reindexing.
- **SwarmNFS now SwarmFS** – *SwarmNFS* is renamed *SwarmFS* to reflect the greater scope of capabilities. The latest version of SwarmFS has improvements for performance and maintainability in NFS integrations with Swarm.
- **Grafana Dashboards** – New versions of public Caringo dashboards are published on grafana.com. The dashboards have links to each other on the top navigation bar:
 - [Caringo Swarm System Monitoring v12.0](#) (cluster view)
 - New: [Caringo Swarm Node View](#) (detailed view)
 - [Caringo Swarm Gateway Monitoring v7](#)
 - [Caringo Videoclipping Dashboard v1.0.1](#)
 - [Caringo Swarm AlertManager v12](#)

Swarm 11 Highlights

Swarm combines the scalable software-defined object storage of Swarm Storage with the components to support diverse implementations:

- *Platform Server* – Node for site-wide management and services
- *Storage Cluster* – Cluster for Swarm storage nodes
- *Elasticsearch* – Cluster for search and historical metrics
- *Content Gateway* – Gateway for cloud-based client access (S3)
- *Storage UI* – Website for storage cluster management
- *Content UI* – Website for cloud content management
- *SwarmFS* – Optional connector for NFS clients

- [Swarm 11.3 – launched August 2020](#)
- [Swarm 11.2 – launched June 2020](#)
- [Swarm 11.1 – launched April 2020](#)
- [Swarm 11.0 – launched September 2019](#)

Swarm 11.3 – launched August 2020

Storage	CSN Platform	ES, Metrics, Search	Gateway	Content UI	Storage UI	SwarmFS
11.3	8.3.2	6.8.6 (5.6.12), 6.3.1, 6.3.1	7.0	6.3	2.4	3.0

- **Swarm Performance** – This release of Swarm Storage enhances both cluster performance and memory management. Cluster shutdowns and startups are faster, and better memory management and support for nodes with limited memory improves Swarm performance under high client loads. This release also includes changes that improve Swarm stability and administration, through better handling of volume removal and hotplugging, smoother rebooting, and stronger logging security.
- **Folder Listing Service** – With Gateway 7.0, folder listing support across Swarm clients (such as SwarmFS and S3) has been both completely rearchitected and also newly centralized within Content Gateway. *Folder listing* is what renders the virtual folders (prefixes) on named Swarm objects (such as `FY2019/Q3/object.jpg`) in to familiar folders on users' file systems. The new service makes full use of Elasticsearch 6 features and is no longer bound by ES listing limits. Centralization means that future listing improvements are easier and faster to roll out.
- **UI Changes for NFS Exports** – Dependency on Elasticsearch is removed for NFS export definitions. The Swarm UI is updated to reflect the less complex NFS definitions.
- **SwarmFS Redesign** – With 3.0, SwarmFS removes dependency on Elasticsearch versioning and makes full use of the new folder listing service in Content Gateway 7.0. The new architecture brings many benefits to SwarmFS implementations, such as centralized authentication through Gateway, improved query security, and freedom to move the Elasticsearch cluster to a more secure network location.

Swarm 11.2 – launched June 2020

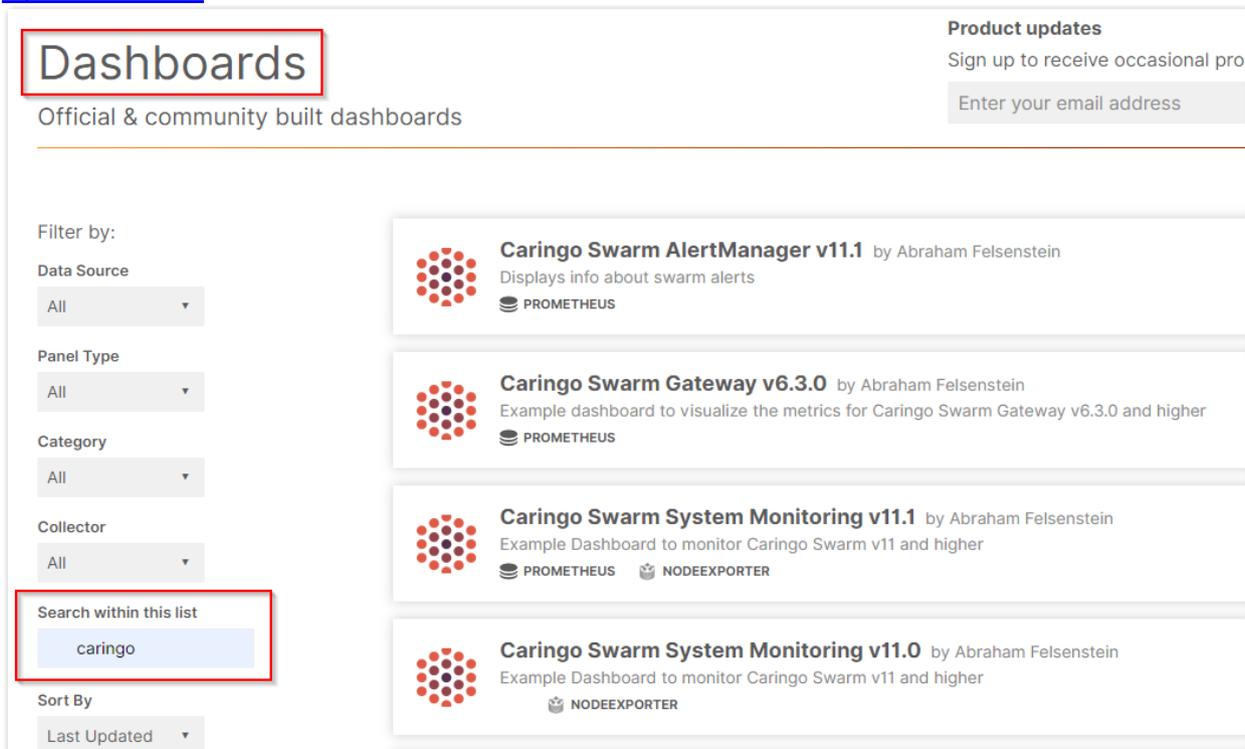
Storage	CSN Platform	ES, Metrics, Search	Gateway	Content UI	Storage UI	SwarmFS
11.2	8.3.2	6.8.6 (5.6.12), 6.3.1, 6.3.1	6.4	6.3	2.3	2.4

- **Next-generation SEND Method** – This release of Swarm Storage and Content Gateway focuses on adding support for the next generation of SCSP SEND, which are foundational to future capabilities. SCSP SEND allows forcing an object to be written immediately another cluster for which a replication feed exists.

Swarm 11.1 – launched April 2020

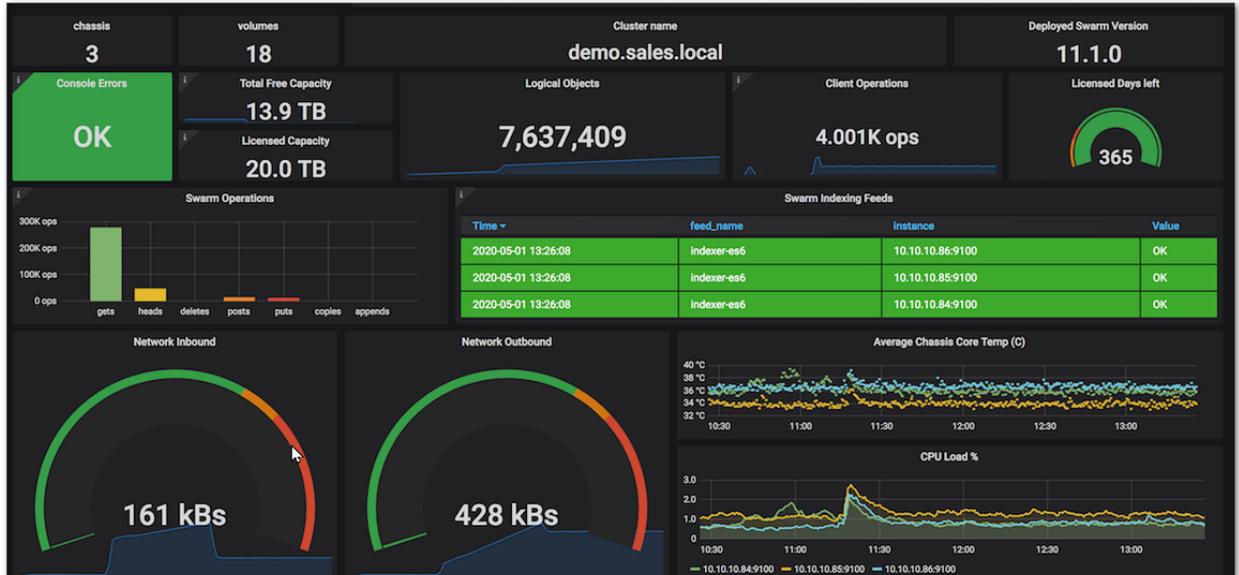
Storage	CSN Platform	ES, Metrics, Search	Gateway	Content UI	Storage UI	SwarmFS
11.1	8.3.2	6.8.6 (5.6.12), 6.3.0, 6.3.0	6.3	6.3	2.3	2.4

- **Grafana Dashboards for Swarm Monitoring** – To offer sophisticated visualization of the Prometheus Node Exporter and related Swarm data, DataCore has published public Grafana dashboards for monitoring Swarm implementations. Search the dashboards for *Caringo* to see all dashboards for Swarm products and features: <https://grafana.com/grafana/dashboards?search=caringo>. See [Prometheus Node Exporter and Grafana](#).

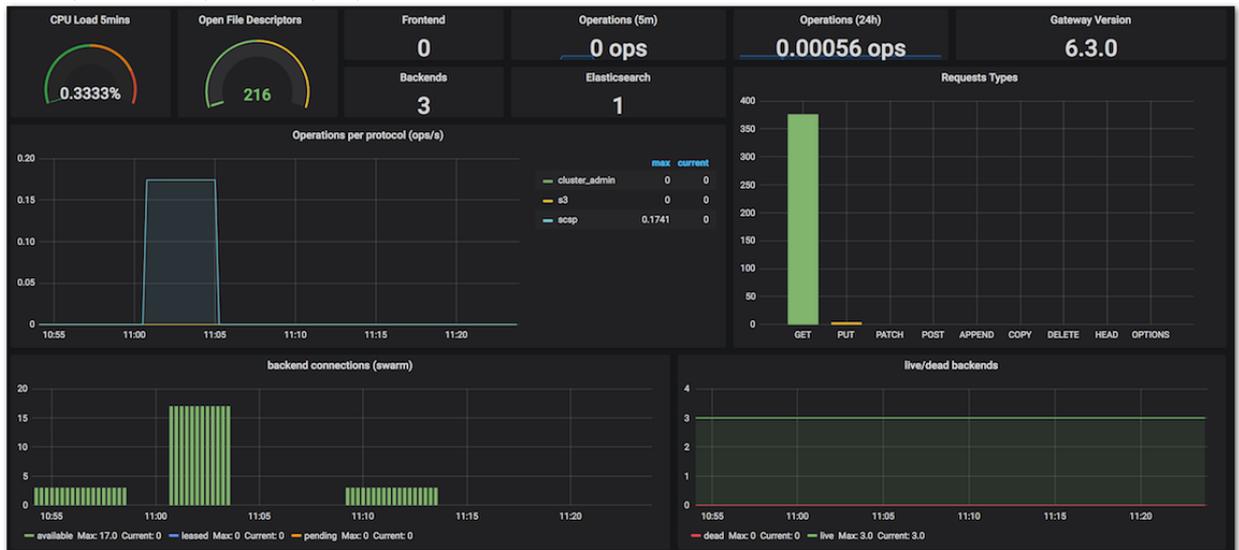


Customized dashboards are available for the following products:

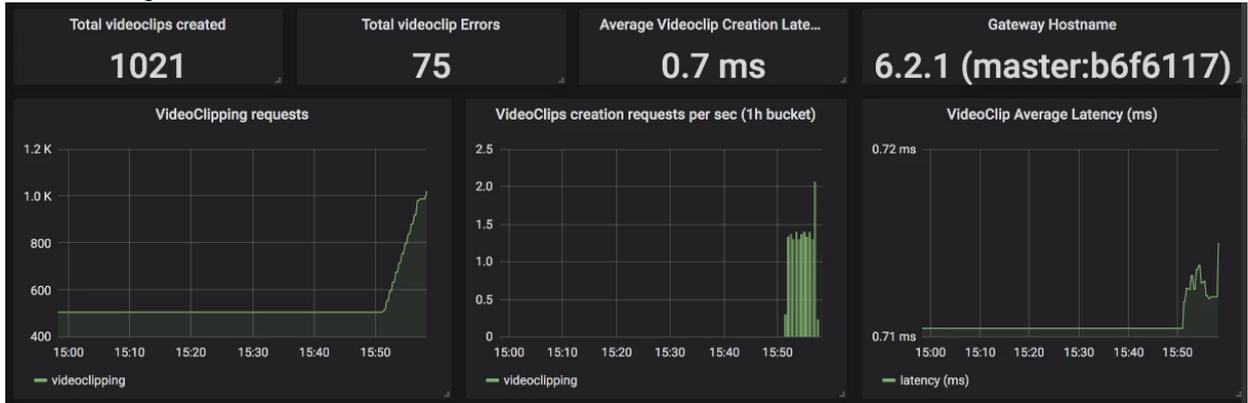
- **Swarm System Monitoring** (separate dashboards for Storage 11.0 and Storage 10.2.) – Covers cluster health, capacity, indexing, licensing, temperature, and network and CPU loads, as well as cluster-wide operations:



- **Gateway Monitoring** (for Gateway 6.3) – covers details of CPU load, operations, connections, and HTTP status codes:



- **Video Clipping** (*optional*, for Gateway 6.2+) – covers numbers, rates, and error counts for video clipping requests (see [Video Clipping for Partial File Restore](#)); errors are counted by stage (*preprocessing, processing, postprocessing*), to help with troubleshooting:



- **Gateway Support for Untenanted Objects** – *Untenanted* objects are unnamed objects that are written to Swarm without specifying a domain. Gateway adds support for untenanted objects, and accepts the Swarm setting `enforceTenancy=false` and provides Content Metering metrics for these objects. Upgrading to Content Gateway is possible if still using SCSP Proxy due to untenanted unnamed objects. Gateway 6.2.0 accepts untenanted objects, so it is a drop-in replacement for SCSP Proxy, which is deprecated. With `enforceTenancy=false`, untenanted objects can continue to be created with existing client applications. Note: untenanted objects are incompatible with the Content UI.
- **Elasticsearch 6** – Swarm supports and ships with Elasticsearch 6, which is a version allowing upgrades-in-place (without reindexing) going forward several releases. Both ES2 and ES5 are deprecated in the next release.
- **Python 3 throughout Swarm** – All Swarm Storage usage of Python 2 is uniformly upgraded to Python 3, which brings with it a small performance boost, up to 20% improvement for high loads.
- **Modernization** – Extensive work has modernized the Linux kernel to Debian 10 and the drivers and components, which allowed for comprehensive updates across Swarm's third-party tools and dependencies.
- **Large cluster support** – This release includes performance improvements for very large clusters, which benefits clusters of all sizes.
- **Faster uploads from Content UI** – To speed the performance of large uploads, the part size for multipart uploads has been increased to 25 MB, which is a common S3 client default part size.

Swarm 11.0 – launched September 2019

Storage	CSN Platform	ES, Metrics, Search	Gateway	Content UI	Storage UI	SwarmFS
11.0	8.3.2	5.6.12, 5.0.8, 5.0.10	6.2	6.2	2.3	2.3

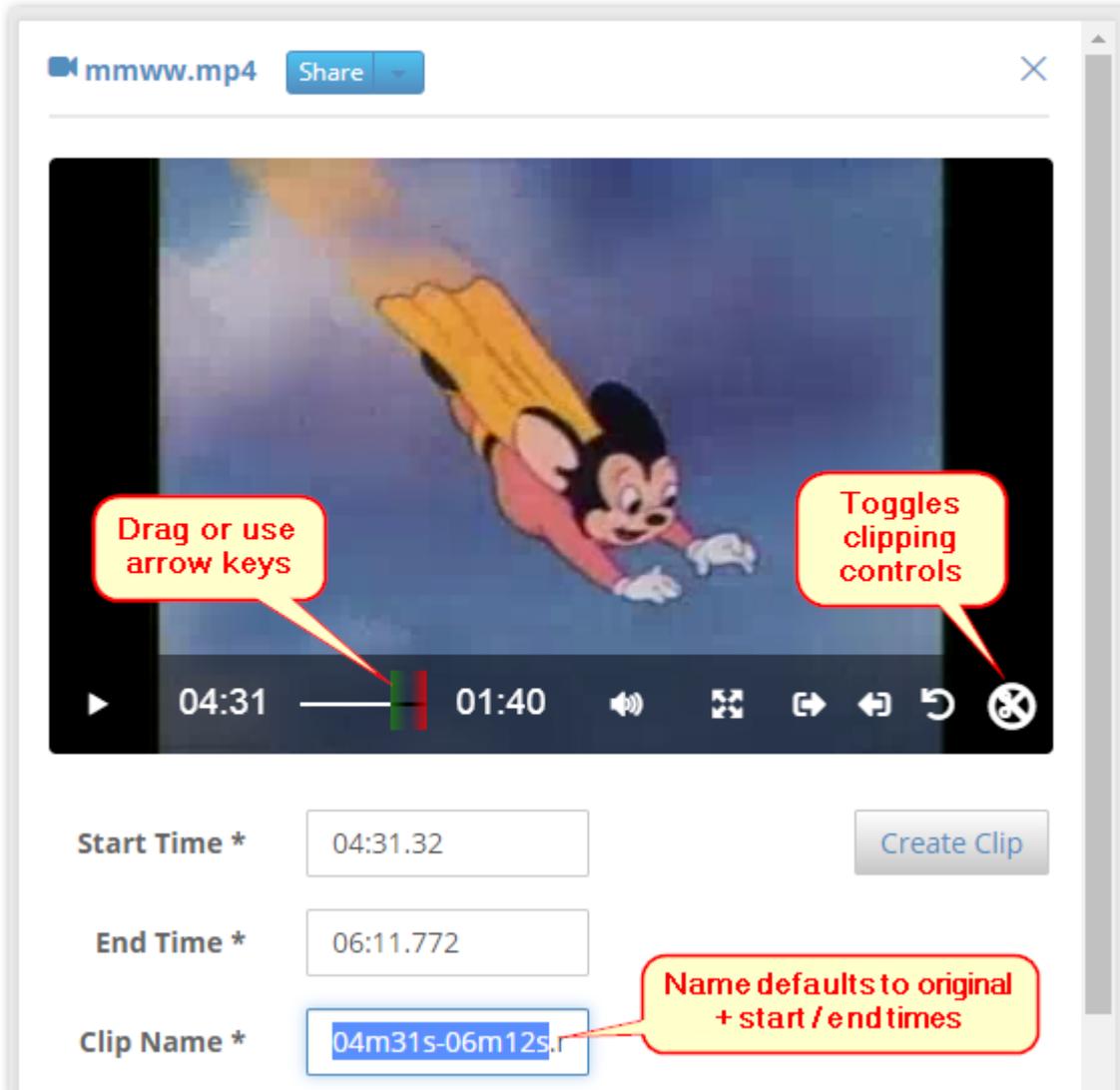
- S3 Backup for DR** – Swarm allows tiering to public cloud services for convenient and affordable off-premises storage for disaster recovery (DR). Content Gateway supports Amazon S3, which has the widest support in the industry, so S3-compatible endpoints are the first cloud destination from Swarm. By implementing an S3 backup feed from Swarm, backups become continuous, have minimal latency, and need minimal intervention and monitoring. The S3 Backup leverages Swarm's mature feed mechanism, which offers long-term iteration over objects in the cluster, proven method for tracking work as it is performed, and mechanisms for TLS connections and forward proxies. Having the parallelism of the entire cluster makes best use of network bandwidth, while sending the backups through a forward proxy enables bandwidth throttling.

The screenshot shows the 'S3 Backup Feed' configuration page. At the top right are 'Revert' and 'Save' buttons. The 'Name' field contains 'S3 backup whole cluster'. The 'Scope' section has 'Entire source cluster (global)' selected. 'Propagate deletes' is checked and 'Enabled'. Under 'Target S3 Provider', the 'Endpoint' is 's3.amazonaws.com', 'Region' is 'US West', and 'Bucket' is 'MyAWSbucket'. The 'Credentials' field contains an access key ID. The 'Use SSL' option is selected as 'Yes'. The 'Port' is '443'. The 'Local Cluster Forward Proxy' option is selected as 'None'. The 'Threads' field is set to '6'. Annotations include a callout for 'S3 access key ID and secret access key' pointing to the credentials field, and another for 'Allows basic throttling control' pointing to the 'Local Cluster Forward Proxy' option.

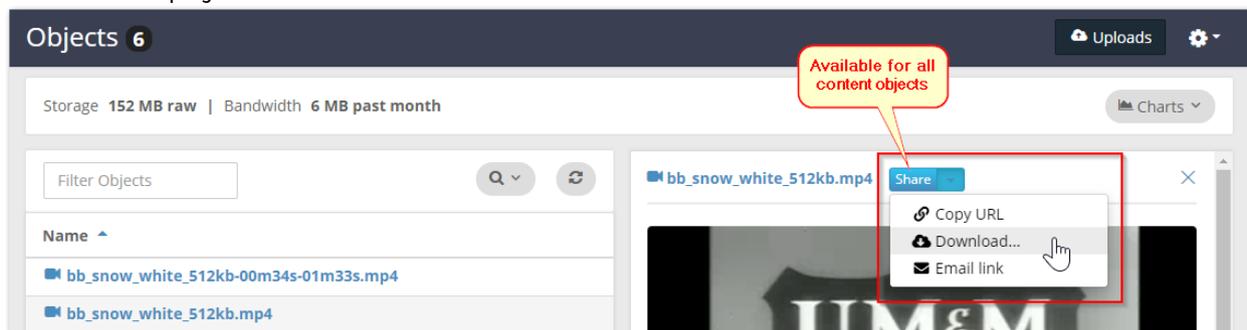
S3 Backup occurs as an integral part of an operating Swarm cluster. After the feed is started, the progress can be monitor with warnings of blockages and particular object failures, as with any other feed. The S3 Backup feed honors the versioning settings in the cluster, as enabled, disabled, or suspended throughout the domains and buckets. The feed keeps the backup current and trimmed: when disabling Swarm versioning on buckets or domains, delete buckets or domains, or have object lifepoints expire, the Swarm feeds mechanism processes the expired content as deleted, allowing the S3 Backup feed to clear them from the S3 bucket.

- S3 Backup Restore** – The Restore tool runs outside of Swarm, using a command-line interface for executing the data and restoration tasks. Restore what is needed: either the entire cluster, or portions. Swarm supports bulk restores at the granularity of cluster, domain, or bucket, as well as more surgical restores of a few objects. Run multiple copies to achieve a faster, parallel recovery.

- Video Clipping / Partial File Restore** – As soon as a video is uploaded into a bucket in Swarm, it is viewable and sharable from the Content UI. With the new **Video Clipping** controls optionally installed, portions can be excerpted out and stored as new, standalone videos within Swarm. The tool saves the clip into the same bucket as the source video, creating a default name that includes the original name and the start and end times of the clip. Each clip created is a standalone video, not a stub pointing to a range in the original; therefore, there is no dependency on the original, which speeds and simplifies distribution.



- Content Sharing** – A new **Share** button appears next to the name when selecting an object in a Content UI listing to view it. The button opens a menu of commands for content sharing, including copying the URL to the local clipboard, downloading the file locally, and opening the default email program to email the link to someone else.



- **Large Uploads through Content UI** – The Content UI file uploader has been redesigned to write directly to Swarm storage and bypass spooling altogether, which removes the prior 4 GB limit. The Content UI accepts more and larger files and is able to recover and resume uploads that encounter errors, and the uploader is compatible with Swarm containerization.
- **Containerization Architecture** – The architecture work of Swarm 10 continues with build-out of support for containerization, so Swarm storage nodes can be managed in containers.
- **Prometheus Node Exporter** – The Prometheus Node Exporter preview has new, global-friendly naming for the node exporter metrics files, and the statistics have richer state information, including node status (idle, mounting, initializing, retiring, ...).
- **Platform IPMI Credential Storage** – Platform Server can store the IPMI username and password for an externally managed Swarm chassis. Credentials do not need to be entered when Platform runs power on/off commands using IPMI over LAN by storing the credentials.
- **SwarmFS Tuning** – The SwarmFS 2.3 release adds several new Advanced settings for tuning SwarmFS behavior and performance in different implementations.
- **S3 Compatibility** – Content Gateway continues to keep pace with the evolving S3 protocol changes to maintain best-in-class compatibility with applications written for the AWS S3 protocol.
- **FileFly 3.1** – The 3.1 release of FileFly features new, generic support for S3-compatible endpoints as well as numerous performance and UI improvements.

Content Gateway Release Notes

Content Gateway is a lightweight, web-scale application used by companies who want to deploy massively scalable, secure, multi-tenant object storage clouds. Its primary components include a Gateway, a Content UI (user web portal), and a Metering service.



Note

Review the changes and upgrade impacts for *each version* since the currently running version if upgrading from a prior version.

- [Content Gateway 7.9 Release](#)
- [Content Gateway 7.8 Release](#)
- [Content Gateway 7.6 Release](#)
- [Content Gateway 7.5 Release](#)
- [Content Gateway 7.4 Release](#)
- [Content Gateway 7.3 Release](#)
- [Content Gateway 7.2 Release](#)
- [Content Gateway 7.1 Release](#)
- [Content Gateway 7.0 Release](#)
- [Content Gateway 6 Release](#)

Content Gateway 7.9 Release

- [New Features](#)
- [Upgrade Impacts](#)
- [Impacts for 7.9](#)
- [Impacts for 7.8](#)
- [Watch Items and Issues](#)

New Features

- [Bucket Lifecycle Policy](#) feature with S3 and SCSP support provides data protection from any malicious deletions and overwrites, without any middle layer. Swarm UI provides an enable/disable feature to apply lifecycle policy on a bucket (within a domain) using a checkbox.

Upgrade Impacts

See [Upgrading Gateway](#), to upgrade from a version of Gateway 6. See [Upgrading from Gateway 5.x](#), if migrating from Elasticsearch 2.3.3 and Gateway 5.

Starting from Gateway 7.8, Elasticsearch 6.8.6 is no longer supported. Remain on Gateway 7.7 until the rolling upgrade is completed from Elasticsearch 6.8.6 to 7.5.2.

Address the upgrade impacts for this *and each prior version* since the currently running version:

Impacts for 7.9

- *Version Requirements*
 - Swarm Storage 14.1.0 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.6

Impacts for 7.8

- *Version Requirements*
 - Swarm Storage 14.0.1 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.5

Fixed in 7.8

S3 delimiter listings now display all objects in a versioned bucket. Prior to Gateway 7.8, some objects are not visible and can cause some S3 client "sync" operations to re-copy objects.

Impacts for 7.5

- *Version Requirements*

- Swarm Storage 12.0.1 or higher
- Elasticsearch 7.5.2
- Content UI 7.4

Impacts for 7.4

- *Version Requirements*
 - Swarm Storage 12.0.1 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.4

Impacts for 7.3

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.3

Impacts for 7.2

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.2

Impacts for 7.1

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2 – Migration to Elasticsearch 6 from either Elasticsearch 2 or 5, with reindexing, must be performed before upgrading. Because the ES 6 database is binary-compatible, upgrade in place to the current version is possible. See [How to Upgrade Swarm](#).
 - Content UI 7.0
- **Password security**
 - The script to initialize Gateway (`/opt/caringo/cloudgateway/bin/initgateway`), a one-time step after installing Gateway, generates the master encryption key that is used in password security for the Gateway configuration and IDSYS files. The first time upgrading from a version prior to 7.1, run this initialization again to enable the feature.
 - If downgrading from 7.1, errors are encountered related to the inability to authenticate using the encrypted passwords in the configuration and IDSYS files. Replace any encrypted credentials with original versions. (CLOUD-3209)

Impacts for 7.0

- *Version Requirements*
 - Swarm Storage 11.2 or higher
 - Elasticsearch 6.8.6 or 5.6.12
 - Content UI 6.3, if used
- Enable the Gateway service manually after upgrading: `systemctl enable cloudgateway`. (CLOUD-3193)

- To support processes requiring repeated bucket PUT requests to succeed, those requests now always return 409 Conflict, regardless of owner, instead of 403 Forbidden for non-owners. This differs from AWS S3 behavior. (CLOUD-3167)

See [Content Gateway 6.4 Release](#) for impacts from prior releases.

Watch Items and Issues

These are known operational limitations that exist for Gateway.

- When using the default RHEL/CentOS configuration of IPTABLES, traffic to the Gateway will be blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s).
- Gateway is not compatible with Linux PAM modules that depend upon interactive validation operations such as OTP or biometric scanners.

See [Content Gateway 6.4 Release](#) for known issues from prior releases that are still applicable, apart from those appearing above as **Fixed**.

Third-Party Components for Gateway 7.9

For licensing information, see [Open Source Software Licenses](#).

org.slf4j:jul-to-slf4j:jar:1.7.30

org.slf4j:slf4j-api:jar:1.7.30

org.apache.logging.log4j:log4j-api:jar:2.16.0

org.apache.logging.log4j:log4j-core:jar:2.16.0

org.apache.logging.log4j:log4j-web:jar:2.16.0

org.apache.logging.log4j:log4j-slf4j-impl:jar:2.16.0

commons-net:commons-net:jar:3.8.0

com.caringo:caringo-util:jar:1.0.13

com.caringo:jscsp:jar:1.2.11-SNAPSHOT

org.apache.httpcomponents:httpclient:jar:4.5.13

org.apache.httpcomponents:httpcore:jar:4.4.13

org.apache.httpcomponents:httpclient-cache:jar:4.5.13

javax.jmdns:jmdns:jar:3.4.1

org.eclipse.jetty:jetty-server:jar:9.4.35.v20201120

javax.servlet:javax.servlet-api:jar:3.1.0

org.eclipse.jetty:jetty-http:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-io:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-servlet:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-security:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-util-ajax:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-servlets:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-continuation:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-util:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-webapp:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-xml:jar:9.4.35.v20201120

org.testng:testng:jar:6.11:test

com.beust:jcommander:jar:1.64:test

org.yaml:snakeyaml:jar:1.17

com.fasterxml.jackson.core:jackson-databind:jar:2.9.10

com.fasterxml.jackson.core:jackson-annotations:jar:2.9.10

com.fasterxml.jackson.core:jackson-core:jar:2.9.10

com.fasterxml.jackson.dataformat:jackson-dataformat-xml:jar:2.9.10

org.codehaus.woodstox:stax2-api:jar:4.2

com.fasterxml.woodstox:woodstox-core:jar:5.3.0

com.fasterxml.jackson.dataformat:jackson-dataformat-yaml:jar:2.9.10

com.fasterxml.jackson.module:jackson-module-jaxb-annotations:jar:2.9.10

com.fasterxml.jackson.jaxrs:jackson-jaxrs-json-provider:jar:2.9.10

com.fasterxml.jackson.jaxrs:jackson-jaxrs-base:jar:2.9.10

com.fasterxml.jackson.datatype:jackson-datatype-jsr310:jar:2.9.10

org.codehaus.woodstox:woodstox-core-asl:jar:4.4.1

javax.xml.stream:stax-api:jar:1.0-2

com.nebhale.jsonpath:jsonpath:jar:1.2

com.github.java-json-tools:json-schema-validator:jar:2.2.14

com.github.java-json-tools:jackson-coreutils-equivalence:jar:1.0

com.github.java-json-tools:jackson-coreutils:jar:2.0

com.github.java-json-tools:msg-simple:jar:1.2

com.github.java-json-tools:btf:jar:1.3

com.github.java-json-tools:json-schema-core:jar:1.2.14

com.github.java-json-tools:uri-template:jar:0.10

org.mozilla:rhino:jar:1.7.7.2

com.sun.mail:mailapi:jar:1.6.2

com.googlecode.libphonenumber:libphonenumber:jar:8.11.1

com.google.code.findbugs:jsr305:jar:3.0.2

net.sf.jopt-simple:jopt-simple:jar:5.0.4

com.google.guava:guava:jar:30.1-jre

com.google.guava:failureaccess:jar:1.0.1
com.google.guava:listenablefuture:jar:9999.0-empty-to-avoid-conflict-with-guava
org.checkerframework:checker-qual:jar:3.5.0
com.google.errorprone:error_prone_annotations:jar:2.3.4
com.google.j2objc:j2objc-annotations:jar:1.3
org.elasticsearch.client:elasticsearch-rest-high-level-client:jar:7.10.2
org.elasticsearch:elasticsearch:jar:7.10.2
org.elasticsearch:elasticsearch-core:jar:7.10.2
org.elasticsearch:elasticsearch-secure-sm:jar:7.10.2
org.elasticsearch:elasticsearch-x-content:jar:7.10.2
com.fasterxml.jackson.dataformat:jackson-dataformat-smile:jar:2.9.10
com.fasterxml.jackson.dataformat:jackson-dataformat-cbor:jar:2.9.10
org.elasticsearch:elasticsearch-geo:jar:7.10.2
org.apache.lucene:lucene-core:jar:8.7.0
org.apache.lucene:lucene-analyzers-common:jar:8.7.0
org.apache.lucene:lucene-backward-codecs:jar:8.7.0
org.apache.lucene:lucene-grouping:jar:8.7.0
org.apache.lucene:lucene-highlighter:jar:8.7.0
org.apache.lucene:lucene-join:jar:8.7.0
org.apache.lucene:lucene-memory:jar:8.7.0
org.apache.lucene:lucene-misc:jar:8.7.0
org.apache.lucene:lucene-queries:jar:8.7.0
org.apache.lucene:lucene-queryparser:jar:8.7.0
org.apache.lucene:lucene-sandbox:jar:8.7.0
org.apache.lucene:lucene-spatial-extras:jar:8.7.0
org.apache.lucene:lucene-spatial3d:jar:8.7.0
org.apache.lucene:lucene-suggest:jar:8.7.0
org.elasticsearch:elasticsearch-cli:jar:7.10.2
com.carrotsearch:hppc:jar:0.8.1

com.tdunning:t-digest:jar:3.2

org.hdrhistogram:HdrHistogram:jar:2.1.9

org.elasticsearch:jna:jar:5.5.0

org.elasticsearch.client:elasticsearch-rest-client:jar:7.10.2

org.apache.httpcomponents:httpasyncclient:jar:4.1.4

org.apache.httpcomponents:httpcore-nio:jar:4.4.12

org.elasticsearch.plugin:mapper-extras-client:jar:7.10.2

org.elasticsearch.plugin:parent-join-client:jar:7.10.2

org.elasticsearch.plugin:aggs-matrix-stats-client:jar:7.10.2

org.elasticsearch.plugin:rank-eval-client:jar:7.10.2

org.elasticsearch.plugin:lang-mustache-client:jar:7.10.2

com.github.spullara.mustache.java:compiler:jar:0.9.6

org.glassfish.jersey.containers:jersey-container-servlet:jar:2.25.1

org.glassfish.jersey.containers:jersey-container-servlet-core:jar:2.25.1

org.glassfish.jersey.core:jersey-common:jar:2.25.1

javax.annotation:javax.annotation-api:jar:1.2

org.glassfish.jersey.bundles.repackaged:jersey-guava:jar:2.25.1

org.glassfish.hk2:hk2-api:jar:2.5.0-b32

org.glassfish.hk2:hk2-utils:jar:2.5.0-b32

org.glassfish.hk2.external:aopalliance-repackaged:jar:2.5.0-b32

org.glassfish.hk2:hk2-locator:jar:2.5.0-b32

org.javassist:javassist:jar:3.20.0-GA

org.glassfish.hk2:osgi-resource-locator:jar:1.0.1

org.glassfish.jersey.core:jersey-server:jar:2.25.1

org.glassfish.jersey.core:jersey-client:jar:2.25.1

org.glassfish.jersey.media:jersey-media-jaxb:jar:2.25.1

[javax.ws.rs:javax.ws.rs-api:jar:2.0.1](#)

org.glassfish.jersey.ext:jersey-bean-validation:jar:2.25.1

org.glassfish.hk2.external:javax.inject:jar:2.5.0-b32

javax.validation:validation-api:jar:1.1.0.Final
org.hibernate:hibernate-validator:jar:5.1.3.Final
org.jboss.logging:jboss-logging:jar:3.1.3.GA
com.fasterxml:classmate:jar:1.0.0
javax.el:javax.el-api:jar:2.2.4
org.glassfish.web:javax.el:jar:2.2.4
javax.enterprise:cdi-api:jar:1.2
javax.interceptor:javax.interceptor-api:jar:1.2
javax.inject:javax.inject:jar:1
org.mockito:mockito-all:jar:1.10.8:test
commons-configuration:commons-configuration:jar:1.10
commons-lang:commons-lang:jar:2.6
commons-logging:commons-logging:jar:1.1.1
commons-validator:commons-validator:jar:1.7
commons-beanutils:commons-beanutils:jar:1.9.4
commons-digester:commons-digester:jar:2.1
commons-collections:commons-collections:jar:3.2.2
org.kohsuke:akuma:jar:1.10
joda-time:joda-time:jar:2.10.9
commons-fileupload:commons-fileupload:jar:1.4
commons-io:commons-io:jar:2.8.0
org.apache.commons:commons-vfs2:jar:2.7.0
org.apache.hadoop:hadoop-hdfs-client:jar:3.3.0
org.kohsuke:libpam4j:jar:1.11
net.java.dev.jna:jna:jar:5.6.0
org.jasig.cas.client:cas-client-core:jar:3.6.1
commons-codec:commons-codec:jar:1.15
org.bouncycastle:bcpkix-jdk15on:jar:1.63
org.bouncycastle:bcprov-jdk15on:jar:1.63

javax.mail:javax.mail-api:jar:1.6.2
com.sun.mail:javax.mail:jar:1.6.2
javax.activation:activation:jar:1.1
com.caringo:storage-mgmt-api:jar:10.0.0
io.swagger:swagger-annotations:jar:1.5.15
com.squareup.okhttp:okhttp:jar:2.7.5
com.squareup.okio:okio:jar:1.6.0
com.squareup.okhttp:logging-interceptor:jar:2.7.5
com.google.code.gson:gson:jar:2.8.1
io.gsonfire:gson-fire:jar:1.8.0
org.threeten:threetenbp:jar:1.3.5
io.prometheus:simpleclient:jar:0.9.0
io.prometheus:simpleclient_hotspot:jar:0.9.0
io.prometheus:simpleclient_servlet:jar:0.9.0
io.prometheus:simpleclient_common:jar:0.9.0
io.prometheus:simpleclient_jetty_jdk8:jar:0.9.0
io.prometheus:simpleclient_jetty:jar:0.9.0
com.onelogin:java-saml:jar:2.5.0
com.onelogin:java-saml-core:jar:2.5.0
org.apache.commons:commons-lang3:jar:3.4
org.apache.santuario:xmlsec:jar:2.1.4
jakarta.xml.bind:jakarta.xml.bind-api:jar:2.3.3
jakarta.activation:jakarta.activation-api:jar:1.2.2

Content Gateway 7.8 Release

- [New Features](#)
- [Upgrade Impacts](#)
- [Impacts for 7.8](#)
- [Watch Items and Issues](#)

New Features

Hybrid Cloud Copy to S3 – Hybrid Cloud Copy to S3 feature in Gateway 7.8 provides the capability to copy objects from Swarm to a target S3 cloud storage destination. Native cloud applications/services running on utility computing can work with data directly from the target cloud storage.

Upgrade Impacts

See [Upgrading Gateway](#) to upgrade from a version of Gateway 6. See [Upgrading from Gateway 5.x](#) if migrating from Elasticsearch 2.3.3 and Gateway 5.

Starting from Gateway 7.8, Elasticsearch 6.8.6 is no longer supported. Remain on Gateway 7.7 until the rolling upgrade is completed from Elasticsearch 6.8.6 to 7.5.2.

Address the upgrade impacts for this *and each prior version* since the currently running version:

Impacts for 7.8

- *Version Requirements*
 - Swarm Storage 14.0.1 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.5

Fixed in 7.8

S3 delimiter listings now display all objects in a versioned bucket. Prior to Gateway 7.8, some objects are not visible and can cause some S3 client “sync” operations to re-copy objects.

Impacts for 7.6

- *Version Requirements*
 - Swarm Storage 12.0.1 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.4

Impacts for 7.5

- *Version Requirements*
 - Swarm Storage 12.0.1 or higher
 - Elasticsearch 7.5.2

- Content UI 7.4

Impacts for 7.4

- *Version Requirements*
 - Swarm Storage 12.0.1 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.4

Impacts for 7.3

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.3

Impacts for 7.2

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.2

Impacts for 7.1

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2 – Migration to Elasticsearch 6 from either Elasticsearch 2 or 5, with reindexing, must be performed before upgrading. Because the ES 6 database is binary-compatible, upgrade in place to the current version is possible. See [How to Upgrade Swarm](#).
 - Content UI 7.0
- **Password security**
 - The script to initialize Gateway (`/opt/caringo/cloudgateway/bin/initgateway`), a one-time step after installing Gateway, generates the master encryption key that is used in password security for the Gateway configuration and IDSYS files. The first time upgrading from a version prior to 7.1, run this initialization again to enable the feature.
 - If downgrading from 7.1, errors are encountered related to the inability to authenticate using the encrypted passwords in the configuration and IDSYS files. Replace any encrypted credentials with original versions. (CLOUD-3209)

Impacts for 7.0

- *Version Requirements*
 - Swarm Storage 11.2 or higher
 - Elasticsearch 6.8.6 or 5.6.12
 - Content UI 6.3, if used
- Enable the Gateway service manually after upgrading: `systemctl enable cloudgateway`. (CLOUD-3193)
- To support processes requiring repeated bucket PUT requests to succeed, those requests now always return 409 Conflict, regardless of owner, instead of 403 Forbidden for non-owners. This differs from AWS S3 behavior. (CLOUD-3167)

See [Content Gateway 6.4 Release](#) for impacts from prior releases.

Watch Items and Issues

These are known operational limitations that exist for Gateway.

- When using the default RHEL/CentOS configuration of IPTABLES, traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s).
- Gateway is not compatible with Linux PAM modules that depend upon interactive validation operations such as OTP or biometric scanners.

See [Content Gateway 6.4 Release](#) for known issues from prior releases that are still applicable, apart from those appearing above as **Fixed**.

Third-Party Components for Gateway 7.8

For licensing information, see [Open Source Software Licenses](#).

org.slf4j:jul-to-slf4j:jar:1.7.30

org.slf4j:slf4j-api:jar:1.7.30

org.apache.logging.log4j:log4j-api:jar:2.16.0

org.apache.logging.log4j:log4j-core:jar:2.16.0

org.apache.logging.log4j:log4j-web:jar:2.16.0

org.apache.logging.log4j:log4j-slf4j-impl:jar:2.16.0

commons-net:commons-net:jar:3.8.0

com.caringo:caringo-util:jar:1.0.13

com.caringo:jscsp:jar:1.2.11-SNAPSHOT

org.apache.httpcomponents:httpclient:jar:4.5.13

org.apache.httpcomponents:httpcore:jar:4.4.13

org.apache.httpcomponents:httpclient-cache:jar:4.5.13

javax.jmdns:jmdns:jar:3.4.1

org.eclipse.jetty:jetty-server:jar:9.4.35.v20201120

javax.servlet:javax.servlet-api:jar:3.1.0

org.eclipse.jetty:jetty-http:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-io:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-servlet:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-security:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-util-ajax:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-servlets:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-continuation:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-util:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-webapp:jar:9.4.35.v20201120

org.eclipse.jetty:jetty-xml:jar:9.4.35.v20201120

org.testng:testng:jar:6.11:test

com.beust:jcommander:jar:1.64:test
org.yaml:snakeyaml:jar:1.17
com.fasterxml.jackson.core:jackson-databind:jar:2.9.10
com.fasterxml.jackson.core:jackson-annotations:jar:2.9.10
com.fasterxml.jackson.core:jackson-core:jar:2.9.10
com.fasterxml.jackson.dataformat:jackson-dataformat-xml:jar:2.9.10
org.codehaus.woodstox:stax2-api:jar:4.2
com.fasterxml.woodstox:woodstox-core:jar:5.3.0
com.fasterxml.jackson.dataformat:jackson-dataformat-yaml:jar:2.9.10
com.fasterxml.jackson.module:jackson-module-jaxb-annotations:jar:2.9.10
com.fasterxml.jackson.jaxrs:jackson-jaxrs-json-provider:jar:2.9.10
com.fasterxml.jackson.jaxrs:jackson-jaxrs-base:jar:2.9.10
com.fasterxml.jackson.datatype:jackson-datatype-jsr310:jar:2.9.10
org.codehaus.woodstox:woodstox-core-asl:jar:4.4.1
javax.xml.stream:stax-api:jar:1.0-2
com.nebhale.jsonpath:jsonpath:jar:1.2
com.github.java-json-tools:json-schema-validator:jar:2.2.14
com.github.java-json-tools:jackson-coreutils-equivalence:jar:1.0
com.github.java-json-tools:jackson-coreutils:jar:2.0
com.github.java-json-tools:msg-simple:jar:1.2
com.github.java-json-tools:btf:jar:1.3
com.github.java-json-tools:json-schema-core:jar:1.2.14
com.github.java-json-tools:uri-template:jar:0.10
org.mozilla:rhino:jar:1.7.7.2
com.sun.mail:mailapi:jar:1.6.2
com.googlecode.libphonenumber:libphonenumber:jar:8.11.1
com.google.code.findbugs:jsr305:jar:3.0.2
net.sf.jopt-simple:jopt-simple:jar:5.0.4
com.google.guava:guava:jar:30.1-jre

com.google.guava:failureaccess:jar:1.0.1

com.google.guava:listenablefuture:jar:9999.0-empty-to-avoid-conflict-with-guava

org.checkerframework:checker-qual:jar:3.5.0

com.google.errorprone:error_prone_annotations:jar:2.3.4

com.google.j2objc:j2objc-annotations:jar:1.3

org.elasticsearch.client:elasticsearch-rest-high-level-client:jar:7.10.2

org.elasticsearch:elasticsearch:jar:7.10.2

org.elasticsearch:elasticsearch-core:jar:7.10.2

org.elasticsearch:elasticsearch-secure-sm:jar:7.10.2

org.elasticsearch:elasticsearch-x-content:jar:7.10.2

com.fasterxml.jackson.dataformat:jackson-dataformat-smile:jar:2.9.10

com.fasterxml.jackson.dataformat:jackson-dataformat-cbor:jar:2.9.10

org.elasticsearch:elasticsearch-geo:jar:7.10.2

org.apache.lucene:lucene-core:jar:8.7.0

org.apache.lucene:lucene-analyzers-common:jar:8.7.0

org.apache.lucene:lucene-backward-codecs:jar:8.7.0

org.apache.lucene:lucene-grouping:jar:8.7.0

org.apache.lucene:lucene-highlighter:jar:8.7.0

org.apache.lucene:lucene-join:jar:8.7.0

org.apache.lucene:lucene-memory:jar:8.7.0

org.apache.lucene:lucene-misc:jar:8.7.0

org.apache.lucene:lucene-queries:jar:8.7.0

org.apache.lucene:lucene-queryparser:jar:8.7.0

org.apache.lucene:lucene-sandbox:jar:8.7.0

org.apache.lucene:lucene-spatial-extras:jar:8.7.0

org.apache.lucene:lucene-spatial3d:jar:8.7.0

org.apache.lucene:lucene-suggest:jar:8.7.0

org.elasticsearch:elasticsearch-cli:jar:7.10.2

com.carrotsearch:hppc:jar:0.8.1

com.tdunning:t-digest:jar:3.2

org.hdrhistogram:HdrHistogram:jar:2.1.9

org.elasticsearch:jna:jar:5.5.0

org.elasticsearch.client:elasticsearch-rest-client:jar:7.10.2

org.apache.httpcomponents:httpasyncclient:jar:4.1.4

org.apache.httpcomponents:httpcore-nio:jar:4.4.12

org.elasticsearch.plugin:mapper-extras-client:jar:7.10.2

org.elasticsearch.plugin:parent-join-client:jar:7.10.2

org.elasticsearch.plugin:aggs-matrix-stats-client:jar:7.10.2

org.elasticsearch.plugin:rank-eval-client:jar:7.10.2

org.elasticsearch.plugin:lang-mustache-client:jar:7.10.2

com.github.spullara.mustache.java:compiler:jar:0.9.6

org.glassfish.jersey.containers:jersey-container-servlet:jar:2.25.1

org.glassfish.jersey.containers:jersey-container-servlet-core:jar:2.25.1

org.glassfish.jersey.core:jersey-common:jar:2.25.1

javax.annotation:javax.annotation-api:jar:1.2

org.glassfish.jersey.bundles.repackaged:jersey-guava:jar:2.25.1

org.glassfish.hk2:hk2-api:jar:2.5.0-b32

org.glassfish.hk2:hk2-utils:jar:2.5.0-b32

org.glassfish.hk2.external:aopalliance-repackaged:jar:2.5.0-b32

org.glassfish.hk2:hk2-locator:jar:2.5.0-b32

org.javassist:javassist:jar:3.20.0-GA

org.glassfish.hk2:osgi-resource-locator:jar:1.0.1

org.glassfish.jersey.core:jersey-server:jar:2.25.1

org.glassfish.jersey.core:jersey-client:jar:2.25.1

org.glassfish.jersey.media:jersey-media-jaxb:jar:2.25.1

[javax.ws.rs:javax.ws.rs-api:jar:2.0.1](#)

org.glassfish.jersey.ext:jersey-bean-validation:jar:2.25.1

org.glassfish.hk2.external:javax.inject:jar:2.5.0-b32

javax.validation:validation-api:jar:1.1.0.Final
org.hibernate:hibernate-validator:jar:5.1.3.Final
org.jboss.logging:jboss-logging:jar:3.1.3.GA
com.fasterxml:classmate:jar:1.0.0
javax.el:javax.el-api:jar:2.2.4
org.glassfish.web:javax.el:jar:2.2.4
javax.enterprise:cdi-api:jar:1.2
javax.interceptor:javax.interceptor-api:jar:1.2
javax.inject:javax.inject:jar:1
org.mockito:mockito-all:jar:1.10.8:test
commons-configuration:commons-configuration:jar:1.10
commons-lang:commons-lang:jar:2.6
commons-logging:commons-logging:jar:1.1.1
commons-validator:commons-validator:jar:1.7
commons-beanutils:commons-beanutils:jar:1.9.4
commons-digester:commons-digester:jar:2.1
commons-collections:commons-collections:jar:3.2.2
org.kohsuke:akuma:jar:1.10
joda-time:joda-time:jar:2.10.9
commons-fileupload:commons-fileupload:jar:1.4
commons-io:commons-io:jar:2.8.0
org.apache.commons:commons-vfs2:jar:2.7.0
org.apache.hadoop:hadoop-hdfs-client:jar:3.3.0
org.kohsuke:libpam4j:jar:1.11
net.java.dev.jna:jna:jar:5.6.0
org.jasig.cas.client:cas-client-core:jar:3.6.1
commons-codec:commons-codec:jar:1.15
org.bouncycastle:bcpkix-jdk15on:jar:1.63
org.bouncycastle:bcprov-jdk15on:jar:1.63

javax.mail:javax.mail-api:jar:1.6.2
com.sun.mail:javax.mail:jar:1.6.2
javax.activation:activation:jar:1.1
com.caringo:storage-mgmt-api:jar:10.0.0
io.swagger:swagger-annotations:jar:1.5.15
com.squareup.okhttp:okhttp:jar:2.7.5
com.squareup.okio:okio:jar:1.6.0
com.squareup.okhttp:logging-interceptor:jar:2.7.5
com.google.code.gson:gson:jar:2.8.1
io.gsonfire:gson-fire:jar:1.8.0
org.threeten:threetenbp:jar:1.3.5
io.prometheus:simpleclient:jar:0.9.0
io.prometheus:simpleclient_hotspot:jar:0.9.0
io.prometheus:simpleclient_servlet:jar:0.9.0
io.prometheus:simpleclient_common:jar:0.9.0
io.prometheus:simpleclient_jetty_jdk8:jar:0.9.0
io.prometheus:simpleclient_jetty:jar:0.9.0
com.onelogin:java-saml:jar:2.5.0
com.onelogin:java-saml-core:jar:2.5.0
org.apache.commons:commons-lang3:jar:3.4
org.apache.santuario:xmlsec:jar:2.1.4
jakarta.xml.bind:jakarta.xml.bind-api:jar:2.3.3
jakarta.activation:jakarta.activation-api:jar:1.2.2

Content Gateway 7.6 Release

- [New Features](#)
- [Upgrade Impacts](#)
- [Watch Items and Issues](#)

New Features

S3 Object Locking – Object locking feature in Gateway 7.6 prevents object versions from being deleted or overwritten – for a fixed amount of time or indefinitely.

Upgrade Impacts

See [Upgrading Gateway](#) to upgrade from a version of Gateway 6. See [Upgrading from Gateway 5.x](#) if migrating from Elasticsearch 2.3.3 and Gateway 5.

Address the upgrade impacts for this *and each prior version* since currently running version:

Impacts for 7.6

- *Version Requirements*
 - Swarm Storage 12.0.1 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.4

Impacts for 7.5

- *Version Requirements*
 - Swarm Storage 12.0.1 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.4

Impacts for 7.4

- *Version Requirements*
 - Swarm Storage 12.0.1 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.4

Impacts for 7.3

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2

- Content UI 7.3

Impacts for 7.2

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.2

Impacts for 7.1

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2 – Migration to Elasticsearch 6 from either Elasticsearch 2 or 5, with reindexing, must be performed before upgrading. Because the ES 6 database is binary-compatible, upgrade in place to the current version is possible. See [How to Upgrade Swarm](#).
 - Content UI 7.0
- **Password security**
 - The script to initialize Gateway (`/opt/caringo/cloudgateway/bin/initgateway`), a one-time step after installing Gateway, generates the master encryption key that is used in password security for the Gateway configuration and IDSYS files. The first time upgrading from a version prior to 7.1, run this initialization again to enable the feature.
 - If downgrading from 7.1, errors are encountered related to the inability to authenticate using the encrypted passwords in the configuration and IDSYS files. Replace any encrypted credentials with original versions. (CLOUD-3209)

Impacts for 7.0

- *Version Requirements*
 - Swarm Storage 11.2 or higher
 - Elasticsearch 6.8.6 or 5.6.12
 - Content UI 6.3, if used
- Enable the Gateway service manually after upgrading: `systemctl enable cloudgateway`. (CLOUD-3193)
- To support processes requiring repeated bucket PUT requests to succeed, those requests now always return 409 Conflict, regardless of owner, instead of 403 Forbidden for non-owners. This differs from AWS S3 behavior. (CLOUD-3167)

See [Content Gateway 6.4 Release](#) for impacts from prior releases.

Watch Items and Issues

These are known operational limitations that exist for Gateway.

- When using the default RHEL/CentOS configuration of IPTABLES, traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s).
- Gateway is not compatible with Linux PAM modules that depend upon interactive validation operations such as OTP or biometric scanners.

See [Content Gateway 6.4 Release](#) for known issues from prior releases that are still applicable, apart from those appearing above as **Fixed**.

Third-Party Components for Gateway 7.6

For licensing information, see [Open Source Software Licenses](#).

Components are unchanged from the prior release: See [Third-Party Components for Gateway 7.5](#).

Content Gateway 7.5 Release

- [Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Issues](#)

Changes

- The `scsp.allowSwarmAdminIP` setting now also accepts CIDR style ranges like "172.30.128/17" and "172.30.128.0/17" in addition to previously accepted values of "all" or a list of IP addresses. (CLOUD-1191)
- Appropriate warnings are logged on startup if the Content Gateway is configured to run in legacy mode, but is also configured to enable services that are not supported in legacy mode (S3, etc.). (CLOUD-3291)
- Improved error handling of a rare error (500 InternalError, ClientProtocolException) by logging details and retrying. If the retry causes a problem, set `debug.retryClientProtocolException = 0`. (CLOUD-3321)
- The `managementPassword` setting is no longer optional and is now required. Always verify `managementUser` and `managementPassword` configured.

Fixed:

- Resolved issues with replication feeds targeting a Content Gateway version 7.1, 7.2, 7.3 or 7.4. Upgrade to Swarm 12.1 and Gateway 7.5 where this is now fixed. (CLOUD-3323)

Upgrade Impacts

See [Upgrading Gateway](#) to upgrade from a version of Gateway 6. See [Upgrading from Gateway 5.x](#) if migrating from Elasticsearch 2.3.3 and Gateway 5.

Address the upgrade impacts for this *and each prior version* since the currently running version:

Impacts for 7.5

- *Version Requirements*
 - Swarm Storage 12.0.1 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.4

Impacts for 7.4

- *Version Requirements*
 - Swarm Storage 12.0.1 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.4

Impacts for 7.3

- *Version Requirements*

- Swarm Storage 12.0 or higher
- Elasticsearch 7.5.2
- Content UI 7.3

Impacts for 7.2

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.2

Impacts for 7.1

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2 – Migration to Elasticsearch 6 from either Elasticsearch 2 or 5, with reindexing, must be performed before upgrading. Because the ES 6 database is binary-compatible, upgrade in place to the current version is possible. See [How to Upgrade Swarm](#).
 - Content UI 7.0
- **Password security**
 - The script to initialize Gateway (`/opt/caringo/cloudgateway/bin/initgateway`), a one-time step after installing Gateway, generates the master encryption key that is used in password security for the Gateway configuration and IDSYS files. The first time upgrading from a version prior to 7.1, run this initialization again to enable the feature.
 - If downgrading from 7.1, errors are encountered related to the inability to authenticate using the encrypted passwords in the configuration and IDSYS files. Replace any encrypted credentials with original versions. (CLOUD-3209)

Impacts for 7.0

- *Version Requirements*
 - Swarm Storage 11.2 or higher
 - Elasticsearch 6.8.6 or 5.6.12
 - Content UI 6.3, if used
- Enable the Gateway service manually after upgrading: `systemctl enable cloudgateway`. (CLOUD-3193)
- To support processes requiring repeated bucket PUT requests to succeed, those requests now always return 409 Conflict, regardless of owner, instead of 403 Forbidden for non-owners. This differs from AWS S3 behavior. (CLOUD-3167)

See [Content Gateway 6.4 Release](#) for impacts from prior releases.

Watch Items and Issues

These are known operational limitations that exist for Gateway.

- When using the default RHEL/CentOS configuration of IPTABLES, traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s).
- Gateway is not compatible with Linux PAM modules that depend upon interactive validation operations such as OTP or biometric scanners.

See [Content Gateway 6.4 Release](#) for known issues from prior releases that are still applicable, apart from those appearing above as **Fixed**.

Third-Party Components for Gateway 7.5

For licensing information, see [Open Source Software Licenses](#).

Components are unchanged from the prior release: See [Third-Party Components for Gateway 7.4](#).

Content Gateway 7.4 Release

- [Changes](#)
- [Known Issues](#)
- [Upgrade Impacts](#)
- [Watch Items and Issues](#)

Changes

- Prometheus/Node exporter support can now be configured (port, enabled/disabled)

Fixed:

- Fixed application compatibility with S3 v2 and v4 signature methods
- Improved startup handling when storage nodes are initially offline

Known Issues

These are known operational limitations that exist for Gateway 7.4.

- **Missing [scsp] allowSwarmAdminIP setting causes startup failure** – Verify the setting [scsp] allowSwarmAdminIP exists in gateway.cfg (even if the value is blank) for Content Gateway 7.4. Content Gateway does not start if the setting is not present. Add the following configuration entry to gateway.cfg to mitigate the risk. (CLOUD-3325)

Workaround for Gateway 7.4 startup failure – gateway.cfg

```
[scsp]
allowSwarmAdminIP=
```

Upgrade Impacts

See [Upgrading Gateway](#) to upgrade from a version of Gateway 6. See [Upgrading from Gateway 5.x](#) if migrating from Elasticsearch 2.3.3 and Gateway 5.

Address the upgrade impacts for this *and each prior version* since the currently running version:

Impacts for 7.4

- *Version Requirements*
 - Swarm Storage 12.0.1 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.4

Impacts for 7.3

- *Version Requirements*

- Swarm Storage 12.0 or higher
- Elasticsearch 7.5.2
- Content UI 7.3

Impacts for 7.2

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.2

Impacts for 7.1

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2 – Migration to Elasticsearch 6 from either Elasticsearch 2 or 5, with reindexing, must be performed before upgrading. Because the ES 6 database is binary-compatible, upgrade in place to the current version is possible. See [How to Upgrade Swarm](#).
 - Content UI 7.0
- **Password security**
 - The script to initialize Gateway (`/opt/caringo/cloudgateway/bin/initgateway`), a one-time step after installing Gateway, generates the master encryption key that is used in password security for the Gateway configuration and IDSYS files. The first time upgrading from a version prior to 7.1, run this initialization again to enable the feature.
 - If downgrading from 7.1, errors are encountered related to the inability to authenticate using the encrypted passwords in the configuration and IDSYS files. Replace any encrypted credentials with original versions. (CLOUD-3209)

Impacts for 7.0

- *Version Requirements*
 - Swarm Storage 11.2 or higher
 - Elasticsearch 6.8.6 or 5.6.12
 - Content UI 6.3, if used
- Enable the Gateway service manually after upgrading: `systemctl enable cloudgateway`. (CLOUD-3193)
- To support processes requiring repeated bucket PUT requests to succeed, those requests now always return 409 Conflict, regardless of owner, instead of 403 Forbidden for non-owners. This differs from AWS S3 behavior. (CLOUD-3167)

See [Content Gateway 6.4 Release](#) for impacts from prior releases.

Watch Items and Issues

These are known operational limitations that exist for Gateway.

- Traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s) when using the default RHEL/CentOS configuration of IPTABLES.
- Gateway is not compatible with Linux PAM modules that depend upon interactive validation operations such as OTP or biometric scanners.

See [Content Gateway 6.4 Release](#) for known issues from prior releases that are still applicable, apart from those appearing above as **Fixed**.

Third-Party Components for Gateway 7.4

For licensing information, see [Open Source Software Licenses](#).

org.apache.commons:commons-lang3:jar:3.4
com.fasterxml.jackson.dataformat:jackson-dataformat-cbor:jar:2.9.10
org.apache.lucene:lucene-sandbox:jar:6.6.1
org.javassist:javassist:jar:3.20.0-GA
org.eclipse.jetty:jetty-security:jar:9.4.35.v20201120
javax.interceptor:javax.interceptor-api:jar:1.2
com.squareup.okio:okio:jar:1.6.0
org.apache.santuario:xmlsec:jar:2.1.4
io.prometheus:simpleclient_servlet:jar:0.9.0
io.prometheus:simpleclient_jetty:jar:0.9.0
org.glassfish.jersey.core:jersey-client:jar:2.25.1
org.testng:testng:jar:6.11:test
com.google.code.findbugs:jsr305:jar:3.0.0
com.fasterxml.jackson.core:jackson-annotations:jar:2.9.10
com.fasterxml.jackson.core:jackson-core:jar:2.9.10
com.google.j2objc:j2objc-annotations:jar:1.3
commons-digester:commons-digester:jar:2.1
com.box:json-schema-validator:jar:2.2.10
org.kohsuke:libpam4j:jar:1.11
commons-io:commons-io:jar:2.8.0
commons-codec:commons-codec:jar:1.15
com.fasterxml.jackson.datatype:jackson-datatype-jsr310:jar:2.9.10
commons-logging:commons-logging:jar:1.1.1
org.elasticsearch.client:elasticsearch-rest-high-level-client:jar:5.6.16
org.apache.lucene:lucene-grouping:jar:6.6.1
com.fasterxml.jackson.module:jackson-module-jaxb-annotations:jar:2.9.10
com.carrotsearch:hppc:jar:0.7.1
javax.validation:validation-api:jar:1.1.0.Final
commons-net:commons-net:jar:3.8.0
org.apache.httpcomponents:httpcore:jar:4.4.13
org.slf4j:jul-to-slf4j:jar:1.7.30
org.glassfish.jersey.containers:jersey-container-servlet:jar:2.25.1
javax.mail:javax.mail-api:jar:1.6.2
com.caringo:jscsp:jar:1.2.9
org.apache.httpcomponents:httpclient:jar:4.5.13
org.glassfish.hk2:hk2-locator:jar:2.5.0-b32
org.threeten:threetenbp:jar:1.3.5
org.apache.httpcomponents:httppasyncclient:jar:4.1.2
org.hdrhistogram:HdrHistogram:jar:2.1.9
org.apache.hadoop:hadoop-hdfs-client:jar:3.3.0
commons-lang:commons-lang:jar:2.6
javax.activation:activation:jar:1.1
org.elasticsearch:jna:jar:4.4.0-1
org.eclipse.jetty:jetty-webapp:jar:9.4.35.v20201120
javax.enterprise:cdi-api:jar:1.2
javax.mail:mailapi:jar:1.4.3
org.jboss.logging:jboss-logging:jar:3.1.3.GA
org.glassfish.jersey.media:jersey-media-jaxb:jar:2.25.1

org.checkerframework:checker-qual:jar:3.5.0
org.apache.lucene:lucene-memory:jar:6.6.1
com.fasterxml.jackson.dataformat:jackson-dataformat-xml:jar:2.9.10
com.sun.mail:javax.mail:jar:1.6.2
commons-beanutils:commons-beanutils:jar:1.9.4
com.beust:jcommander:jar:1.64:test
org.locationtech.spatial4j:spatial4j:jar:0.6
javax.inject:javax.inject:jar:1
javax.servlet:javax.servlet-api:jar:3.1.0
org.glassfish.hk2:osgi-resource-locator:jar:1.0.1
org.glassfish.jersey.ext:jersey-bean-validation:jar:2.25.1
org.mockito:mockito-all:jar:1.10.8:test
org.eclipse.jetty:jetty-server:jar:9.4.35.v20201120
com.fasterxml.jackson.dataformat:jackson-dataformat-smile:jar:2.9.10
org.eclipse.jetty:jetty-util:jar:9.4.35.v20201120
javax.xml.stream:stax-api:jar:1.0-2
org.eclipse.jetty:jetty-continuation:jar:9.4.35.v20201120
org.apache.commons:commons-vfs2:jar:2.7.0
net.sf.jopt-simple:jopt-simple:jar:4.6
org.bouncycastle:bcprov-jdk15on:jar:1.63
io.gsonfire:gson-fire:jar:1.8.0
org.glassfish.jersey.bundles.repackaged:jersey-guava:jar:2.25.1
com.tdunning:t-digest:jar:3.0
com.vividolutions:jts:jar:1.13
com.google.guava:listenablefuture:jar:9999.0-empty-to-avoid-conflict-with-guava
org.glassfish.web:javax.el:jar:2.2.4
com.fasterxml.jackson.core:jackson-databind:jar:2.9.10
com.fasterxml.woodstox:woodstox-core:jar:5.3.0
com.fasterxml.jackson.jaxrs:jackson-jaxrs-base:jar:2.9.10
io.swagger:swagger-annotations:jar:1.5.15
org.eclipse.jetty:jetty-util-ajax:jar:9.4.35.v20201120
org.eclipse.jetty:jetty-io:jar:9.4.35.v20201120
org.eclipse.jetty:jetty-xml:jar:9.4.35.v20201120
javax.jmdns:jmdns:jar:3.4.1
org.apache.lucene:lucene-queryparser:jar:6.6.1
com.googlecode.libphonenumber:libphonenumber:jar:6.2
org.apache.lucene:lucene-spatial:jar:6.6.1
org.yaml:snakeyaml:jar:1.17
org.kohsuke:akuma:jar:1.10
org.glassfish.hk2:hk2-api:jar:2.5.0-b32
org.apache.lucene:lucene-spatial-extras:jar:6.6.1
com.google.guava:failureaccess:jar:1.0.1
org.apache.lucene:lucene-backward-codecs:jar:6.6.1
org.apache.lucene:lucene-suggest:jar:6.6.1
[javax.ws.rs:javax.ws.rs-api:jar:2.0.1](#)
org.codehaus.woodstox:woodstox-core-asl:jar:4.4.1
net.java.dev.jna:jna:jar:5.6.0
io.prometheus:simpleclient_jetty_jdk8:jar:0.9.0
org.apache.logging.log4j:log4j-api:jar:2.14.0
org.apache.logging.log4j:log4j-slf4j-impl:jar:2.14.0
org.slf4j:slf4j-api:jar:1.7.30
org.apache.lucene:lucene-misc:jar:6.6.1
com.github.fge:btf:jar:1.2
com.google.code.gson:gson:jar:2.8.1

org.elasticsearch.plugin:aggs-matrix-stats-client:jar:5.6.16
org.bouncycastle:bcpkix-jdk15on:jar:1.63
org.elasticsearch:securesm:jar:1.2
io.prometheus:simpleclient_hotspot:jar:0.9.0
com.github.fge:jackson-coreutils:jar:1.8
io.prometheus:simpleclient_common:jar:0.9.0
com.box:json-schema-core:jar:1.2.8
com.github.fge:msg-simple:jar:1.1
org.glassfish.hk2:hk2-utils:jar:2.5.0-b32
org.apache.logging.log4j:log4j-core:jar:2.14.0
io.prometheus:simpleclient:jar:0.9.0
org.glassfish.jersey.core:jersey-common:jar:2.25.1
org.glassfish.jersey.core:jersey-server:jar:2.25.1
com.fasterxml:classmate:jar:1.0.0
joda-time:joda-time:jar:2.10.9
commons-configuration:commons-configuration:jar:1.10
com.google.errorprone:error_prone_annotations:jar:2.3.4
commons-validator:commons-validator:jar:1.7
org.apache.lucene:lucene-core:jar:6.6.1
org.apache.httpcomponents:httpcore-nio:jar:4.4.5
com.nebhale.jsonpath:jsonpath:jar:1.2
org.apache.lucene:lucene-queries:jar:6.6.1
com.onelogin:java-saml-core:jar:2.5.0
com.fasterxml.jackson.jaxrs:jackson-jaxrs-json-provider:jar:2.9.10
com.google.guava:guava:jar:30.1-jre
commons-fileupload:commons-fileupload:jar:1.4
com.caringo:caringo-util:jar:1.0.13
org.apache.lucene:lucene-highlighter:jar:6.6.1
org.jasig.cas.client:cas-client-core:jar:3.6.1
org.elasticsearch.plugin:parent-join-client:jar:5.6.16
org.apache.logging.log4j:log4j-web:jar:2.14.0
org.glassfish.hk2.external:aopalliance-repackaged:jar:2.5.0-b32
org.glassfish.hk2.external:javassist:jar:2.5.0-b32
org.elasticsearch:elasticsearch:jar:5.6.16
org.hibernate.hibernate-validator:jar:5.1.3.Final
org.eclipse.jetty:jetty-servlet:jar:9.4.35.v20201120
javax.annotation:javax.annotation-api:jar:1.2
com.caringo:storage-mgmt-api:jar:10.0.0
org.apache.lucene:lucene-join:jar:6.6.1
org.apache.lucene:lucene-analyzers-common:jar:6.6.1
com.squareup.okhttp:logging-interceptor:jar:2.7.5
com.onelogin:java-saml:jar:2.5.0
commons-collections:commons-collections:jar:3.2.2
org.glassfish.jersey.containers:jersey-container-servlet-core:jar:2.25.1
com.squareup.okhttp:okhttp:jar:2.7.5
com.github.fge:uri-template:jar:0.9
org.apache.lucene:lucene-spatial3d:jar:6.6.1
org.mozilla:rhino:jar:1.7R4
org.elasticsearch.client:elasticsearch-rest-client:jar:5.6.16
javax.el:javax.el-api:jar:2.2.4
org.apache.httpcomponents:httpclient-cache:jar:4.5.13
org.eclipse.jetty:jetty-servlets:jar:9.4.35.v20201120
org.codehaus.woodstox:stax2-api:jar:4.2
org.eclipse.jetty:jetty-http:jar:9.4.35.v20201120

com.fasterxml.jackson.dataformat:jackson-dataformat-yaml:jar:2.9.10

Content Gateway 7.3 Release

- [Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Issues](#)

Changes

- Adds management features for long-term SCSP clients using unnamed objects outside of a domain, automatically organizing them into a virtual System domain. Replaces the need for SCSPproxy and unifies metering and access management across all storage domains including the System domain. (CLOUD-3285)

Fixed:

- Folder listings in Content Portal are no longer capped at 2000 entries. (CLOUD-3274)

Upgrade Impacts

See [Upgrading Gateway](#) to upgrade from a version of Gateway 6. See [Upgrading from Gateway 5.x](#) if migrating from Elasticsearch 2.3.3 and Gateway 5.

Address the upgrade impacts for this *and each prior version* since the currently running version:

Impacts for 7.3

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.3

Impacts for 7.2

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.2

Impacts for 7.1

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2 – Migration to Elasticsearch 6 from either Elasticsearch 2 or 5, with reindexing, must be performed before upgrading. Because the ES 6 database is binary-compatible, upgrade in place to the current version is possible. See [How to Upgrade Swarm](#).
 - Content UI 7.0
- **Password security**

- The script to initialize Gateway (`/opt/caringo/cloudgateway/bin/initgateway`), a one-time step after installing Gateway, generates the master encryption key that is used in password security for the Gateway configuration and IDSYS files. The first time upgrading from a version prior to 7.1, run this initialization again to enable the feature.
- If downgrading from 7.1, errors are encountered related to the inability to authenticate using the encrypted passwords in the configuration and IDSYS files. Replace any encrypted credentials with original versions. (CLOUD-3209)

Impacts for 7.0

- *Version Requirements*
 - Swarm Storage 11.2 or higher
 - Elasticsearch 6.8.6 or 5.6.12
 - Content UI 6.3, if used
- Enable the Gateway service manually after upgrading: `systemctl enable cloudgateway`. (CLOUD-3193)
- To support processes requiring repeated bucket PUT requests to succeed, those requests now always return 409 Conflict, regardless of owner, instead of 403 Forbidden for non-owners. This differs from AWS S3 behavior. (CLOUD-3167)

See [Content Gateway 6.4 Release](#) for impacts from prior releases.

Watch Items and Issues

These are known operational limitations that exist for Gateway.

- When using the default RHEL/CentOS configuration of IPTABLES, traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s).
- Gateway is not compatible with Linux PAM modules that depend upon interactive validation operations such as OTP or biometric scanners.

These are known issues in this release:

- The `cloudgateway_audit.log` shows a false HTTP 500 response for SCSP (`format=json`) listing requests. This can be ignored; the client receives an HTTP 200 OK with the correct list results. (CLOUD-3201)

See [Content Gateway 6.4 Release](#) for known issues from prior releases that are still applicable, apart from those appearing above as **Fixed**.

Third-Party Components for Gateway 7.3

For licensing information, see [Open Source Software Licenses](#).

com.beust:jcommander:jar:1.64
com.box:json-schema-core:jar:1.2.8
com.box:json-schema-validator:jar:2.2.10
com.caringo:caringo-util:jar:1.0.12
com.caringo:jscsp:jar:1.2.8
com.caringo:storage-mgmt-api:jar:10.0.0
com.carrotsearch:hppc:jar:0.7.1
com.fasterxml.jackson.core:jackson-annotations:jar:2.9.10
com.fasterxml.jackson.core:jackson-core:jar:2.9.10
com.fasterxml.jackson.core:jackson-databind:jar:2.9.10
com.fasterxml.jackson.dataformat:jackson-dataformat-cbor:jar:2.9.10
com.fasterxml.jackson.dataformat:jackson-dataformat-smile:jar:2.9.10
com.fasterxml.jackson.dataformat:jackson-dataformat-xml:jar:2.9.10
com.fasterxml.jackson.dataformat:jackson-dataformat-yaml:jar:2.9.10
com.fasterxml.jackson.datatype:jackson-datatype-jsr310:jar:2.9.10
com.fasterxml.jackson.jaxrs:jackson-jaxrs-base:jar:2.9.10
com.fasterxml.jackson.jaxrs:jackson-jaxrs-json-provider:jar:2.9.10
com.fasterxml.jackson.module:jackson-module-jaxb-annotations:jar:2.9.10
com.fasterxml.woodstox:woodstox-core:jar:5.3.0
com.fasterxml.classmate:jar:1.0.0
com.github.fge:btf:jar:1.2
com.github.fge:jackson-coreutils:jar:1.8
com.github.fge:msg-simple:jar:1.1
com.github.fge:uri-template:jar:0.9
com.google.code.findbugs:jsr305:jar:3.0.0
com.google.code.gson:gson:jar:2.8.1
com.google.errorprone:error_prone_annotations:jar:2.3.4
com.google.guava:failureaccess:jar:1.0.1
com.google.guava:guava:jar:30.1-jre
com.google.guava:listenablefuture:jar:9999.0-empty-to-avoid-conflict-with-guava
com.google.j2objc:j2objc-annotations:jar:1.3
com.googlecode.libphonenumber:libphonenumber:jar:6.2
com.nebhale.jsonpath:jsonpath:jar:1.2
com.onelogin:java-saml-core:jar:2.5.0
com.onelogin:java-saml:jar:2.5.0
com.squareup.okhttp:logging-interceptor:jar:2.7.5
com.squareup.okhttp:okhttp:jar:2.7.5
com.squareup.okio:okio:jar:1.6.0
com.sun.mail:javax.mail:jar:1.6.2
com.tdunning:t-digest:jar:3.0
com.vividsolutions:jts:jar:1.13
commons-beanutils:commons-beanutils:jar:1.9.4
commons-codec:commons-codec:jar:1.15
commons-collections:commons-collections:jar:3.2.2
commons-configuration:commons-configuration:jar:1.10
commons-digester:commons-digester:jar:2.1
commons-fileupload:commons-fileupload:jar:1.4
commons-io:commons-io:jar:2.8.0

commons-lang:commons-lang:jar:2.6
commons-logging:commons-logging:jar:1.1.1
commons-validator:commons-validator:jar:1.7
io.gsonfire:gson-fire:jar:1.8.0
io.prometheus:simpleclient:jar:0.9.0
io.prometheus:simpleclient_common:jar:0.9.0
io.prometheus:simpleclient_hotspot:jar:0.9.0
io.prometheus:simpleclient_jetty:jar:0.9.0
io.prometheus:simpleclient_jetty_jdk8:jar:0.9.0
io.prometheus:simpleclient_servlet:jar:0.9.0
io.swagger:swagger-annotations:jar:1.5.15
javax.activation:activation:jar:1.1
javax.annotation:javax.annotation-api:jar:1.2
javax.el:javax.el-api:jar:2.2.4
javax.enterprise:cdi-api:jar:1.2
javax.inject:javax.inject:jar:1
javax.interceptor:javax.interceptor-api:jar:1.2
javax.jmdns:jmdns:jar:3.4.1
javax.mail:javax.mail-api:jar:1.6.2
javax.mail:mailapi:jar:1.4.3
javax.servlet:javax.servlet-api:jar:3.1.0
javax.validation:validation-api:jar:1.1.0.Final
[javax.ws.rs:javax.ws.rs-api:jar:2.0.1](#)
javax.xml.stream:stax-api:jar:1.0-2
joda-time:joda-time:jar:2.10.9
net.java.dev.jna:jna:jar:5.6.0
net.sf.jopt-simple:jopt-simple:jar:4.6
org.apache.commons:commons-lang3:jar:3.4
org.apache.commons:commons-vfs2:jar:2.7.0
org.apache.hadoop:hadoop-hdfs-client:jar:3.3.0
org.apache.httpcomponents:httppasynclient:jar:4.1.2
org.apache.httpcomponents:httpclient-cache:jar:4.5.13
org.apache.httpcomponents:httpclient:jar:4.5.13
org.apache.httpcomponents:httpcore-nio:jar:4.4.5
org.apache.httpcomponents:httpcore:jar:4.4.13
org.apache.logging.log4j:log4j-api:jar:2.14.0
org.apache.logging.log4j:log4j-core:jar:2.14.0
org.apache.logging.log4j:log4j-slf4j-impl:jar:2.14.0
org.apache.logging.log4j:log4j-web:jar:2.14.0
org.apache.lucene:lucene-analyzers-common:jar:6.6.1
org.apache.lucene:lucene-backward-codecs:jar:6.6.1
org.apache.lucene:lucene-core:jar:6.6.1
org.apache.lucene:lucene-grouping:jar:6.6.1
org.apache.lucene:lucene-highlighter:jar:6.6.1
org.apache.lucene:lucene-join:jar:6.6.1
org.apache.lucene:lucene-memory:jar:6.6.1
org.apache.lucene:lucene-misc:jar:6.6.1
org.apache.lucene:lucene-queries:jar:6.6.1
org.apache.lucene:lucene-queryparser:jar:6.6.1
org.apache.lucene:lucene-sandbox:jar:6.6.1
org.apache.lucene:lucene-spatial-extras:jar:6.6.1
org.apache.lucene:lucene-spatial3d:jar:6.6.1
org.apache.lucene:lucene-spatial:jar:6.6.1
org.apache.lucene:lucene-suggest:jar:6.6.1

org.apache.santuario.xmlsec:jar:2.1.4
org.bouncycastle.bcpkix-jdk15on:jar:1.63
org.bouncycastle.bcprov-jdk15on:jar:1.63
org.checkerframework:checker-qual:jar:3.5.0
org.codehaus.woodstox.stax2-api:jar:4.2
org.codehaus.woodstox:woodstox-core-asl:jar:4.4.1
org.eclipse.jetty.jetty-continuation:jar:9.4.35.v20201120
org.eclipse.jetty.jetty-http:jar:9.4.35.v20201120
org.eclipse.jetty.jetty-io:jar:9.4.35.v20201120
org.eclipse.jetty.jetty-security:jar:9.4.35.v20201120
org.eclipse.jetty.jetty-server:jar:9.4.35.v20201120
org.eclipse.jetty.jetty-servlet:jar:9.4.35.v20201120
org.eclipse.jetty.jetty-servlets:jar:9.4.35.v20201120
org.eclipse.jetty.jetty-util-ajax:jar:9.4.35.v20201120
org.eclipse.jetty.jetty-util:jar:9.4.35.v20201120
org.eclipse.jetty.jetty-webapp:jar:9.4.35.v20201120
org.eclipse.jetty.jetty-xml:jar:9.4.35.v20201120
org.elasticsearch.client:elasticsearch-rest-client:jar:5.6.16
org.elasticsearch.client:elasticsearch-rest-high-level-client:jar:5.6.16
org.elasticsearch.plugin:aggs-matrix-stats-client:jar:5.6.16
org.elasticsearch.plugin:parent-join-client:jar:5.6.16
org.elasticsearch:elasticsearch:jar:5.6.16
org.elasticsearch:jna:jar:4.4.0-1
org.elasticsearch:securesm:jar:1.2
org.glassfish.hk2.external:aopalliance-repackaged:jar:2.5.0-b32
org.glassfish.hk2.external:javax.inject:jar:2.5.0-b32
org.glassfish.hk2:hk2-api:jar:2.5.0-b32
org.glassfish.hk2:hk2-locator:jar:2.5.0-b32
org.glassfish.hk2:hk2-utils:jar:2.5.0-b32
org.glassfish.hk2:osgi-resource-locator:jar:1.0.1
org.glassfish.jersey.bundles.repackaged:jersey-guava:jar:2.25.1
org.glassfish.jersey.containers:jersey-container-servlet-core:jar:2.25.1
org.glassfish.jersey.containers:jersey-container-servlet:jar:2.25.1
org.glassfish.jersey.core:jersey-client:jar:2.25.1
org.glassfish.jersey.core:jersey-common:jar:2.25.1
org.glassfish.jersey.core:jersey-server:jar:2.25.1
org.glassfish.jersey.ext:jersey-bean-validation:jar:2.25.1
org.glassfish.jersey.media:jersey-media-jaxb:jar:2.25.1
org.glassfish.web:javax.el:jar:2.2.4
org.hdrhistogram:HdrHistogram:jar:2.1.9
org.hibernate.hibernate-validator:jar:5.1.3.Final
org.jasig.cas.client:cas-client-core:jar:3.6.1
org.javassist:javassist:jar:3.20.0-GA
org.jboss.logging:jboss-logging:jar:3.1.3.GA
org.kohsuke.akuma:jar:1.10
org.kohsuke.libpam4j:jar:1.11
org.locationtech.spatial4j:spatial4j:jar:0.6
org.mockito:mockito-all:jar:1.10.8
org.mozilla.rhino:jar:1.7R4
org.slf4j:jul-to-slf4j:jar:1.7.30
org.slf4j:slf4j-api:jar:1.7.30
org.testng:jar:6.11
org.threeten.threetenbp:jar:1.3.5
org.yaml:snakeyaml:jar:1.17

Content Gateway 7.2 Release

- [Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Issues](#)

Changes

This release includes hardening around authorization as well as these improvements in [single sign-on with SAML](#):

- Gateway now supports SAML on the tenant level, allowing for organization-specific SAML authentication in multi-tenant implementations. (CLOUD-3239)
- Gateway has better token handling on SAML logouts. (CLOUD-3245)

Fixed:

- Invalid methods on an SCSP request returned 400 Bad Request instead of the expected 405 Method Not Allowed responses. (CLOUD-3228)

Upgrade Impacts

To upgrade from a version of Gateway 6, see [Upgrading Gateway](#). If you are migrating from Elasticsearch 2.3.3 and Gateway 5, see [Upgrading from Gateway 5.x](#).

Address the upgrade impacts for this *and each prior version* since the currently running version:

Impacts for 7.2

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2
 - Content UI 7.2

Impacts for 7.1

- *Version Requirements*
 - Swarm Storage 12.0 or higher
 - Elasticsearch 7.5.2 – Migration to Elasticsearch 6 from either Elasticsearch 2 or 5, with reindexing, must be performed before upgrading. Because the ES 6 database is binary-compatible, upgrade in place to the current version is possible. See [How to Upgrade Swarm](#).
 - Content UI 7.0
- **Password security**
 - The script to initialize Gateway (`/opt/caringo/cloudgateway/bin/initgateway`), a one-time step after installing Gateway, generates the master encryption key that is used in password security for the Gateway configuration and IDSYS files. The first time upgrading from a version prior to 7.1, run this initialization again to enable the feature.
 - If downgrading from 7.1, errors are encountered related to the inability to authenticate using the encrypted passwords in the configuration and IDSYS files. Replace any encrypted credentials with original versions. (CLOUD-3209)

Impacts for 7.0

- *Version Requirements*
 - Swarm Storage 11.2 or higher
 - Elasticsearch 6.8.6 or 5.6.12
 - Content UI 6.3, if used
- Enable the Gateway service manually after upgrading: `systemctl enable cloudgateway`. (CLOUD-3193)
- To support processes requiring repeated bucket PUT requests to succeed, those requests now always return 409 Conflict, regardless of owner, instead of 403 Forbidden for non-owners. This differs from AWS S3 behavior. (CLOUD-3167)

See [Content Gateway 6.4 Release](#) for impacts from prior releases.

Watch Items and Issues

These are known operational limitations that exist for Gateway.

- When using the default RHEL/CentOS configuration of IPTABLES, traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s).
- Gateway is not compatible with Linux PAM modules that depend upon interactive validation operations such as OTP or biometric scanners.

These are known issues in this release:

- The `cloudgateway_audit.log` shows a false HTTP 500 response for SCSP (`format=json`) listing requests. This can be ignored; the client receives an HTTP 200 OK with the correct list results. (CLOUD-3201)

See [Content Gateway 6.4 Release](#) for known issues from prior releases that are still applicable, apart from those appearing above as **Fixed**.

Third-Party Components for Gateway 7.2

For licensing information, see [Open Source Software Licenses](#).

Components are unchanged from the prior release: See [Third-Party Components for Gateway 7.1](#).

Content Gateway 7.1 Release

- [Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Issues](#)

Changes

Setting for faster GETs – To improve performance through Gateway, enable the new Swarm Storage 12.0 setting `s_csp.enableVolumeRedirects`. This setting permits Gateway to perform redirects of GET requests to volume processes, for greater efficiency, especially with reading small objects. (CLOUD-3205)

Support for folder listings in UIs – With version 7, *folder listing* support across Swarm clients (such as SwarmFS and S3) has been rearchitected and newly centralized within Content Gateway. These folders allow users to interact with bucket objects in an intuitive hierarchical organization. See [Using Virtual Folders](#).

SAML integration for SSO – Gateway now supports SSO (single sign-on) with third-party identity providers using the SAML 2 standard. By implementing SAML, users log in to Swarm browser components (Swarm UI and Content UI) using existing credentials from another source, such as OneLogin, Okta, or Google. See [Enabling SSO with SAML](#). (CLOUD-2970)

Support for larger S3 bucket listings – A new `[storage_cluster]` setting, `indexerSocketTimeout`, allows controlling a timeout affecting the ability to list larger buckets. The value now defaults to 120 seconds. Increase the load balancer (such as HAProxy) "timeout server" and S3 client timeouts as needed to match this. (CLOUD-3171)

Cross-domain cookies – A new `[gateway]` setting, `cookieDomains`, allows the Content UI to use the same authentication token across multiple storage domains that share a common base domain. Gateway does this by using the base domain in place of the request's domain for the `Set-Cookie` response header. (CLOUD-2789)

Password encryption – Gateway now encrypts passwords that are stored `gateway.cfg` and `IDSYS` files. When needing to change management passwords, enter new ones and restart Gateway, which replaces those strings with encrypted versions as part of its startup. (CLOUD-3209)

Easier log levels – For quicker access during troubleshooting, the `logLevel` property is now located at the top of the `logging.yaml` file. (CLOUD-3176)

Fixed:

- Recent rclone releases can make multiple PUT bucket requests fail with a 409 Conflict message. (CLOUD-3213)
- After upgrading, the Gateway service needed to be enabled manually. (CLOUD-3193)

Upgrade Impacts

To upgrade from a version of Gateway 6, see [Upgrading Gateway](#). If migrating from Elasticsearch 2.3.3 and Gateway 5, see [Upgrading from Gateway 5.x](#).

Address the upgrade impacts for this *and each prior version* since the version being upgraded from:

Impacts for 7.1

- *Version Requirements*
 - Swarm Storage 12.0 or higher

- Elasticsearch 7.5.2 – Migration to Elasticsearch 6 from either Elasticsearch 2 or 5, with reindexing, must be performed before upgrading. Because the ES 6 database is binary-compatible, upgrade in place to the current version is possible. See [How to Upgrade Swarm](#).
- Content UI 7.0
- **Password security**
 - The script to initialize Gateway (`/opt/caringo/cloudgateway/bin/initgateway`), a one-time step after installing Gateway, generates the master encryption key that is used in password security for the Gateway configuration and IDSYS files. The first time upgrading from a version prior to 7.1, run this initialization again to enable the feature.
 - If downgrading from 7.1, errors are encountered related to the inability to authenticate using the encrypted passwords in the configuration and IDSYS files. Replace any encrypted credentials with original versions. (CLOUD-3209)

Impacts for 7.0

- *Version Requirements*
 - Swarm Storage 11.2 or higher
 - Elasticsearch 6.8.6 or 5.6.12
 - Content UI 6.3, if used
- Enable the Gateway service manually after upgrading: `systemctl enable cloudgateway`. (CLOUD-3193)
- To support processes requiring repeated bucket PUT requests to succeed, those requests now always return 409 Conflict, regardless of owner, instead of 403 Forbidden for non-owners. This differs from AWS S3 behavior. (CLOUD-3167)

See [Content Gateway 6.4 Release](#) for impacts from prior releases.

Watch Items and Issues

These are known operational limitations that exist for Gateway.

- When using the default RHEL/CentOS configuration of IPTABLES, traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s).
- Gateway is not compatible with Linux PAM modules that depend upon interactive validation operations such as OTP or biometric scanners.

These are known issues in this release:

- Invalid methods on an SCSP request return 400 Bad Request instead of the expected 405 Method Not Allowed responses. (CLOUD-3228)

See [Content Gateway 6.4 Release](#) for known issues from prior releases that are still applicable, apart from those appearing above as **Fixed**.

Third-Party Components for Gateway 7.1

For licensing information, see [Open Source Software Licenses](#).

com.beust: jcommander: jar: 1.64
com.box: json-schema-core: jar: 1.2.6
com.box: json-schema-validator: jar: 2.2.8
com.caringo: caringo-util: jar: 1.0.11
com.caringo: jscsp: jar: 1.2.7
com.caringo: storage-mgmt-api: jar: 10.0.0
com.carrotsearch: hppc: jar: 0.7.1
com.fasterxml.jackson.core: jackson-annotations: jar: 2.9.0
com.fasterxml.jackson.core: jackson-core: jar: 2.9.3
com.fasterxml.jackson.core: jackson-databind: jar: 2.9.3
com.fasterxml.jackson.dataformat: jackson-dataformat-cbor: jar: 2.9.3
com.fasterxml.jackson.dataformat: jackson-dataformat-smile: jar: 2.9.3
com.fasterxml.jackson.dataformat: jackson-dataformat-xml: jar: 2.9.3
com.fasterxml.jackson.dataformat: jackson-dataformat-yaml: jar: 2.9.3
com.fasterxml.jackson.datatype: jackson-datatype-jsr310: jar: 2.9.3
com.fasterxml.jackson.jaxrs: jackson-jaxrs-base: jar: 2.9.3
com.fasterxml.jackson.jaxrs: jackson-jaxrs-json-provider: jar: 2.9.3
com.fasterxml.jackson.module: jackson-module-jaxb-annotations: jar: 2.9.3
com.fasterxml.woodstox: woodstox-core: jar: 5.0.3
com.fasterxml: classmate: jar: 1.0.0
com.github.fge: btf: jar: 1.2
com.github.fge: jackson-coreutils: jar: 1.8
com.github.fge: msg-simple: jar: 1.1
com.github.fge: uri-template: jar: 0.9
com.google.code.findbugs: jsr305: jar: 3.0.0
com.google.code.gson: gson: jar: 2.8.1
com.google.guava: guava: jar: 21.0
com.googlecode.libphonenumber: libphonenumber: jar: 6.2
com.nebhale.jsonpath: jsonpath: jar: 1.2
com.onelogin: java-saml-core: jar: 2.5.0
com.onelogin: java-saml: jar: 2.5.0
com.squareup.okhttp: logging-interceptor: jar: 2.7.5
com.squareup.okhttp: okhttp: jar: 2.7.5
com.squareup.okio: okio: jar: 1.6.0
com.tdunning: t-digest: jar: 3.0
com.vivid solutions: jts: jar: 1.13
commons-beanutils: commons-beanutils: jar: 1.8.3
commons-codec: commons-codec: jar: 1.12
commons-configuration: commons-configuration: jar: 1.10
commons-digester: commons-digester: jar: 1.8
commons-fileupload: commons-fileupload: jar: 1.3.1
commons-io: commons-io: jar: 2.4
commons-lang: commons-lang: jar: 2.6
commons-logging: commons-logging: jar: 1.1.1
commons-validator: commons-validator: jar: 1.4.0
io.gsonfire: gson-fire: jar: 1.8.0
io.prometheus: simpleclient: jar: 0.6.0
io.prometheus: simpleclient_common: jar: 0.6.0

io.prometheus: simpleclient_hotspot: jar: 0.6.0
io.prometheus: simpleclient_jetty: jar: 0.6.0
io.prometheus: simpleclient_jetty_jdk8: jar: 0.6.0
io.prometheus: simpleclient_servlet: jar: 0.6.0
io.swagger: swagger-annotations: jar: 1.5.15
javax.activation: activation: jar: 1.1
javax.annotation: javax.annotation-api: jar: 1.2
javax.el: javax.el-api: jar: 2.2.4
javax.enterprise: cdi-api: jar: 1.2
javax.inject: javax.inject: jar: 1
javax.interceptor: javax.interceptor-api: jar: 1.2
javax.jmdns: jmdns: jar: 3.4.1
javax.mail: mail: jar: 1.4.3
javax.mail: mailapi: jar: 1.4.3
javax.servlet: javax.servlet-api: jar: 3.1.0
javax.validation: validation-api: jar: 1.1.0.Final
[javax.ws.rs](#): [javax.ws.rs](#)-api: jar: 2.0.1
javax.xml.stream: stax-api: jar: 1.0-2
joda-time: joda-time: jar: 2.8.2
net.java.dev.jna: jna: jar: 4.5.2
net.sf.jopt-simple: jopt-simple: jar: 4.6
org.apache.commons: commons-lang3: jar: 3.4
org.apache.commons: commons-vfs2: jar: 2.0
org.apache.httpcomponents: httpasyncclient: jar: 4.1.2
org.apache.httpcomponents: httpclient-cache: jar: 4.5.5
org.apache.httpcomponents: httpclient: jar: 4.5.5
org.apache.httpcomponents: httpcore-nio: jar: 4.4.5
org.apache.httpcomponents: httpcore: jar: 4.4.9
org.apache.logging.log4j: log4j-api: jar: 2.11.1
org.apache.logging.log4j: log4j-core: jar: 2.11.1
org.apache.logging.log4j: log4j-slf4j-impl: jar: 2.11.1
org.apache.logging.log4j: log4j-web: jar: 2.11.1
org.apache.lucene: lucene-analyzers-common: jar: 6.6.1
org.apache.lucene: lucene-backward-codecs: jar: 6.6.1
org.apache.lucene: lucene-core: jar: 6.6.1
org.apache.lucene: lucene-grouping: jar: 6.6.1
org.apache.lucene: lucene-highlighter: jar: 6.6.1
org.apache.lucene: lucene-join: jar: 6.6.1
org.apache.lucene: lucene-memory: jar: 6.6.1
org.apache.lucene: lucene-misc: jar: 6.6.1
org.apache.lucene: lucene-queries: jar: 6.6.1
org.apache.lucene: lucene-queryparser: jar: 6.6.1
org.apache.lucene: lucene-sandbox: jar: 6.6.1
org.apache.lucene: lucene-spatial-extras: jar: 6.6.1
org.apache.lucene: lucene-spatial3d: jar: 6.6.1
org.apache.lucene: lucene-spatial: jar: 6.6.1
org.apache.lucene: lucene-suggest: jar: 6.6.1
org.apache.maven.scm: maven-scm-api: jar: 1.4
org.apache.maven.scm: maven-scm-provider-svn-commons: jar: 1.4
org.apache.maven.scm: maven-scm-provider-svnexe: jar: 1.4
org.apache.santuario: xmlsec: jar: 2.1.4
org.codehaus.plexus: plexus-utils: jar: 1.5.6
org.codehaus.woodstox: stax2-api: jar: 3.1.4
org.codehaus.woodstox: woodstox-core-asl: jar: 4.4.1

org.eclipse.jetty:jetty-continuation:jar:9.4.7.v20170914
org.eclipse.jetty:jetty-http:jar:9.4.7.v20170914
org.eclipse.jetty:jetty-io:jar:9.4.7.v20170914
org.eclipse.jetty:jetty-security:jar:9.4.7.v20170914
org.eclipse.jetty:jetty-server:jar:9.4.7.v20170914
org.eclipse.jetty:jetty-servlet:jar:9.4.7.v20170914
org.eclipse.jetty:jetty-servlets:jar:9.4.7.v20170914
org.eclipse.jetty:jetty-util:jar:9.4.7.v20170914
org.eclipse.jetty:jetty-webapp:jar:9.4.7.v20170914
org.eclipse.jetty:jetty-xml:jar:9.4.7.v20170914
org.elasticsearch.client:elasticsearch-rest-client:jar:5.6.16
org.elasticsearch.client:elasticsearch-rest-high-level-client:jar:5.6.16
org.elasticsearch.plugin:aggs-matrix-stats-client:jar:5.6.16
org.elasticsearch.plugin:parent-join-client:jar:5.6.16
org.elasticsearch:elasticsearch:jar:5.6.16
org.elasticsearch:jna:jar:4.4.0-1
org.elasticsearch:securesm:jar:1.2
org.glassfish.hk2.external:aopalliance-repackaged:jar:2.5.0-b32
org.glassfish.hk2.external:javassist:jar:2.5.0-b32
org.glassfish.hk2:hk2-api:jar:2.5.0-b32
org.glassfish.hk2:hk2-locator:jar:2.5.0-b32
org.glassfish.hk2:hk2-utils:jar:2.5.0-b32
org.glassfish.hk2:osgi-resource-locator:jar:1.0.1
org.glassfish.jersey.bundles.repackaged:jersey-guava:jar:2.25.1
org.glassfish.jersey.containers:jersey-container-servlet-core:jar:2.25.1
org.glassfish.jersey.containers:jersey-container-servlet:jar:2.25.1
org.glassfish.jersey.core:jersey-client:jar:2.25.1
org.glassfish.jersey.core:jersey-common:jar:2.25.1
org.glassfish.jersey.core:jersey-server:jar:2.25.1
org.glassfish.jersey.ext:jersey-bean-validation:jar:2.25.1
org.glassfish.jersey.media:jersey-media-jaxb:jar:2.25.1
org.glassfish.web:javassist-el:jar:2.2.4
org.hdrhistogram:HdrHistogram:jar:2.1.9
org.hibernate:hibernate-validator:jar:5.1.3.Final
org.jasig.cas.client:cas-client-core:jar:3.4.1
org.javassist:javassist:jar:3.20.0-GA
org.jboss.logging:jboss-logging:jar:3.1.3.GA
org.kohsuke:akuma:jar:1.9
org.kohsuke:libpam4j:jar:1.11
org.locationtech.spatial4j:spatial4j:jar:0.6
org.mockito:mockito-all:jar:1.10.8
org.mozilla:rhino:jar:1.7R4
org.slf4j:jul-to-slf4j:jar:1.7.25
org.slf4j:slf4j-api:jar:1.7.25
org.testng:testng:jar:6.11
org.threeten:threetenbp:jar:1.3.5
org.yaml:snakeyaml:jar:1.17
regexp:regexp:jar:1.3

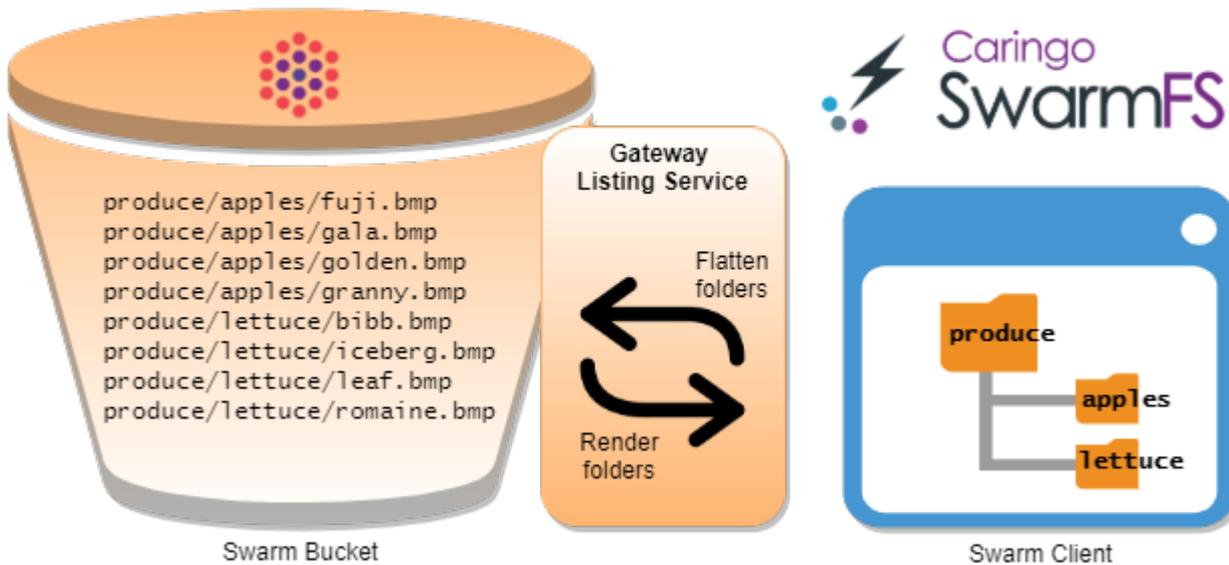
Content Gateway 7.0 Release

- [Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Issues](#)

Changes

Elasticsearch 6 Transition – Gateway 7.0 completes support for Elasticsearch 6. Migration from either Elasticsearch 2 or 5 should be performed at this time. Because the Elasticsearch 6 database is binary-compatible with Elasticsearch 7, upgrading without a migration is possible. Reindexing the cluster is required.

Swarm Folder Listing – With version 7.0, *folder listing* support across Swarm clients (such as SwarmFS and S3) has been both completely rearchitected and also newly centralized within Content Gateway. Folder listing allows Swarm clients to render virtual folders *below* the bucket level of Swarm Storage: it translates any delimited prefixes in Swarm object names (such as in `FY2019/Q3/object.jpg`) into client-side file system folders. These folders allow users to interact with objects in an intuitive hierarchical organization.



The new architecture brings many benefits across Swarm implementations:

- All legacy client-specific code is replaced with a central, unified, and improved approach. Centralization means future listing improvements are easier and faster to roll out.
- The pagination of large listing results is no longer bound to the Elasticsearch limit (`index.max_result_window`).
- The listing service is optimized for features new to Elasticsearch 6.

The scope of this release does not include unnamed objects, caching, folder locking/leasing, or client notification of namespace changes.

Upgrade Impacts

To upgrade from a version of Gateway 6, see [Upgrading Gateway](#). See [Upgrading from Gateway 5.x](#) if migrating from Elasticsearch 2.3.3 and Gateway 5.

Address the upgrade impacts for this *and each prior version* since the currently running version:

Impacts for 7.0

- *Version Requirements*
 - Swarm Storage 11.2 or higher
 - Elasticsearch 6.8.6 or 5.6.12
 - Content UI 6.3, if used
- Enable the Gateway service manually after upgrading: `systemctl enable cloudgateway`. (CLOUD-3193)
- To support processes requiring repeated bucket PUT requests to succeed, those requests now always return 409 Conflict, regardless of owner, instead of 403 Forbidden for non-owners. This differs from AWS S3 behavior. (CLOUD-3167)

See [Content Gateway 6.4 Release](#) for impacts from prior releases.

Watch Items and Issues

These are known operational limitations and watch items existing for Gateway.

- Traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s) when using the default RHEL/CentOS configuration of IPTABLES, .
- Gateway is not compatible with Linux PAM modules depending upon interactive validation operations such as OTP or biometric scanners.

The following are known issues in this release.

- Recent rclone releases can make multiple PUT bucket requests fail (409 Conflict). Workaround: Add `no_check_bucket=true` to the rclone config, or use `"rclone copy --s3-no-check-bucket ..."`. (CLOUD-3213)
- The AWS S3 SDK for C# does not properly sign S3-compatible requests with spaces in the name unless the domain contains ".s3." or ".s3-". See [S3 request signing broken for S3-compatible services](#). (CLOUD-3068)
- The x-amz-storage-class header is not preserved when buckets are created. (CLOUD-3062)
- During new object creation as part of renaming with `?newname`, Gateway does not verify the user has permission to create the new object name (although it is highly likely, because it is a write within the same context). (CLOUD-2966)
- An `s3cmd` or rclone server-side copy request may time out on a multipart copy for >5GB objects (`s4cmd` performs it correctly). Workaround: After verifying it is not the HTTPS proxy timing out, increase the client timeout: set `s3cmd socket_timeout = 600` in `~/.s3cfg` or use `rclone copy --timeout=10m --contimeout=2m caringo:mybucket/5gb caringo:mybucket/subfolder/`. (CLOUD-2949)
- Listings with `max-keys` may be shorter than expected because `CommonPrefixes` are included in the count of keys returned. (CLOUD-2917)
- Usernames are case-insensitive, but listings exclude a token if the username (`myadmin`) does not match the case used when the token was created (`myAdmin`). (CLOUD-2837)
- Multipart PUT requests via recent Cyberduck versions fail with 403 `SignatureDoesNotMatch` when using AWS Signature Version 4. Install the Caringo `.cyberduckprofiles` from [Using the Cyberduck application with Content Gateway S3](#) which force V2 signatures. (CLOUD-2799)
- The policy fails to take effect without warning if a policy document includes a `Principal` with plural "users" or "groups" instead of "user" or "group". (CLOUD-2783)
- 403 S3 V4 Signature mismatch errors may result when using Cyberduck with the "pound" proxy in front of Gateway S3. Workaround: Disable the `Expect` header in the Cyberduck preferences, or (recommended) use a different proxy such as "haproxy". (CLOUD-2628)
- Errors may erroneously report being related to Storage nodes when Gateway cannot connect to Elasticsearch nodes. (CLOUD-2595)
- Because of issues with Range and ETag header handling, video playback of .mp4 streams may not work correctly when served via the Gateway S3 port. It does work when served via the Gateway SCSP port. (CLOUD-1964)
- Gateway caches the Swarm version from the "Server:" response header, so after upgrading Swarm restart Gateway to consistently see the new version. (CLOUD-1271)

- Gateway responds with a 500 (Internal Server Error) instead of 400 (Bad Request) if the size of the metadata headers sent to Swarm is too large. (CLOUD-800)
- The S3 bucket listing StorageClass response element always reports STANDARD. (CLOUD-766)
- The Gateway audit log escapes the "%" characters used by the client as escape characters if an S3 client escapes URI path characters such as "/". URI audit log processing for S3 clients requires double-unescaping when this occurs. (CLOUD-703)

Content Gateway 6 Release

- [Content Gateway 6.1 Release](#)
- [Content Gateway 6.3 Release](#)
- [Content Gateway 6.4 Release](#)
- [Content Gateway 6.2 Release](#)
- [Content Gateway 6.0 Release](#)

Content Gateway 6.1 Release

- [Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Known Issues](#)

Changes

This release of Content Gateway includes these new features:

- Gateway now uses the Version 2 implementation of AWS S3 GET Bucket (List Objects). With this change, Gateway supports the upcoming Docker Distribution 2.7 (registry). (CLOUD-3050)
- Gateway has added infrastructural support for future dynamic features and extensions, such as an upcoming video clip creation tool. (CLOUD-3083, CLOUD-3084)
- Gateway has continuing S3 protocol enhancements to track with Amazon S3 changes.

This release includes these fixes:

- Corrected an issue that prevented periodic flushing of metering records to Elasticsearch. (6.1.1: CLOUD-3097)
- Quota states are not properly evaluated at all times. (CLOUD-3079)

Upgrade Impacts

To upgrade from a version of Gateway 6, see [Upgrading Gateway](#). If you are migrating from Elasticsearch 2.3.3 and are ready to upgrade from Gateway 5, see [Upgrading from Gateway 5.x](#).

Address the upgrade impacts for each of the versions since the one you are currently running:

Impacts for 6.1

- *Version Requirements*
 - Swarm Storage 10.2 or higher
 - Content UI 6.1, if used

Impacts for 6.0

- *Version Requirements*
 - RHEL/CentOS 7: Support for RHEL/CentOS 6 is deprecated; complete the transition to RHEL/CentOS 7 when [upgrading to Elasticsearch 5.6](#).
 - Swarm Storage 10.0 or higher.
 - Elasticsearch 5.6, with a 5.6 search index that is built on the new schema. *Do not upgrade Gateway until the 5.6 search index is complete.* See [Migrating from Older Elasticsearch](#) and [Upgrading from Gateway 5.x](#).
 - ExpanDrive users: version 6.1.0 or higher. (CLOUD-2746)
- **New logging** – For Gateway system and audit logging, review the new, default `logging.yaml` file for any customizations to be implement. See [Gateway Configuration](#).

- Buckets named "_admin" are no longer accessible via Gateway. If a legacy _admin bucket from csmeter exists, remove it with a DELETE request directly to Swarm. (CLOUD-3025)

Watch Items and Known Issues

These are known operational limitations and watch items that exist for Gateway.

- When using the default RHEL/CentOS configuration of IPTABLES, traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s).
- Gateway is not compatible with the fingerprint scanner module for Linux PAM. If it is installed, remove it by running:


```
yum remove fprindt-pam
```
- SCSP reading operations that request a Content-MD5 hash validation and for which there is a hash mismatch causes a storage node to be temporarily removed for the Gateway's connection pool due to the way Swarm reports a hash validation failure.
- Swarm Integrity Seal upgrades cannot be performed through Gateway. They may be done directly to the back-end Swarm cluster.
- If the HTTP cache control headers `If-Modified-Since` and `If-Unmodified-Since` are used, review the discussion of these in the [Storage SCSP Development](#).

The following are known issues in this release.

- The AWS S3 SDK for C# does not properly sign S3-compatible requests with spaces in the name unless the domain contains ".s3." or ".s3-". See <https://github.com/aws/aws-sdk-net/issues/933>. CLOUD-3068
- When buckets are created, the `x-amz-storage-class` header is not preserved. CLOUD-3062
- The Gateway error "Failed reading from client" on a PUT due to "EofException: Early EOF" may occur when clients do not send the full body. This may point to a bug in the client's retry logic, such as not resetting the position marker to the beginning of the file or part. CLOUD-3010
- During new object creation as part of renaming with `?newname`, Gateway does not verify the user has permission to create the new object name (although it is highly likely, because it is a write within the same context). CLOUD-2966
- An `s3cmd` or `rclone` server-side copy request may time out on a multipart copy for >5GB objects (`s4cmd` performs it correctly). Workaround: After you verify it is not the HTTPS proxy timing out, increase the client timeout: `set s3mcd socket_timeout = 600` in `~/s3cfg` or use `rclone copy --timeout=10m --contimeout=2m caringo:mybucket/5gb caringo:mybucket/subfolder/`. CLOUD-2949
- Listings with `max-keys` may be shorter than expected because `CommonPrefixes` are included in the count of keys returned. CLOUD-2917
- Uploading files / photos using Panic's Transmit app on iOS fails due to a 403 Invalid Signature error. CLOUD-2886
- Gateway 5.2.2 and earlier do not output the `NextMarker` field in S3 listings, which can cause some S3 clients such as Caringo Drive, `rclone`, and Transmit to show only 1000 files in a directory or to miss some subdirectories. CLOUD-2871
- Usernames are case-insensitive, but listings exclude a token if the username (`myadmin`) does not match the case used when the token was created (`myAdmin`). CLOUD-2837
- Multipart PUT requests via recent Cyberduck versions fail with 403 SignatureDoesNotMatch when using AWS Signature Version 4. Install the Caringo `.cyberduckprofiles` from [Using the Cyberduck application with Content Gateway S3](#) which force V2 signatures. CLOUD-2799
- If a policy document includes a Principal that has plural "users" or "groups" instead of "user" or "group", the policy fails to take effect without warning. CLOUD-2783
- Versioning-enabled buckets with large numbers of objects may generate Gateway server.log warnings that can be safely ignored: "S3BucketRequestHandler: WARNING: problem with versioned bucket listing. Number of CommonPrefix (2000) exceeds max-size limit (1000)." CLOUD-2643
- 403 S3 V4 Signature mismatch errors may result when using Cyberduck with the "pound" proxy in front of Gateway S3. Workaround: Disable the Expect header in the Cyberduck preferences, or (recommended) use a different proxy such as [HAProxy](#). CLOUD-2628
- When Gateway cannot connect to Elasticsearch nodes, the errors may erroneously report this as being related to Storage nodes. CLOUD-2595
- Because of issues with Range and ETag header handling, video playback of .mp4 streams may not work correctly when served via the Gateway S3 port. It does work when served via the Gateway SCSP port. CLOUD-1964
- Gateway caches the Swarm version from the "Server:" response header, so after upgrading Swarm you must restart Gateway to consistently see the new version. CLOUD-1271

- Gateway responds with a 500 (Internal Server Error) instead of 400 (Bad Request) if the size of the metadata headers sent to Swarm is too large. CLOUD-800
- The S3 bucket listing StorageClass response element always reports STANDARD. CLOUD-766
- If an S3 client escapes URI path characters such as "/", the Gateway audit log escapes the "%" characters used by the client as escape characters. URI audit log processing for S3 clients require double-unescaping when this occurs. CLOUD-703

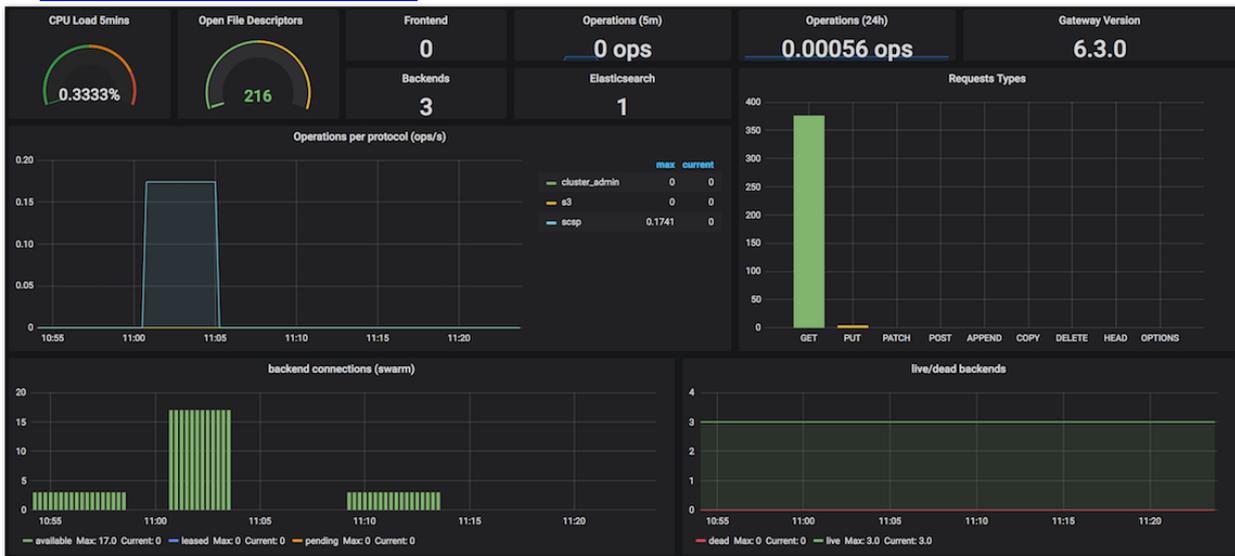
Content Gateway 6.3 Release

- [Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Known Issues](#)

Changes

Elasticsearch 6 – This release of Gateway adds support for Elasticsearch 6.8.6 while maintaining compatibility with Elasticsearch 5.6.12. See [Migrating from Older Elasticsearch](#). If also running SwarmFS, upgrade it to 2.4 after migrating to Elasticsearch 6. (CLOUD-3131)

New Grafana Dashboard – The Grafana dashboard for Gateway 6.3 covers details of CPU load, operations, connections, and HTTP status codes. See [Prometheus Node Exporter and Grafana](#).



Faster Clipping – Reworked connection handling has improved the performance of video clip processing. (CLOUD-3147)

S3 Compatibility

- Multipart writes are long-running operations with initial and final responses. For S3 compatibility, the initial response now returns `x-amz-version-id` with the value of the expected ETag. If there is an error completing the write, there is no new object, and the expected ETag given is not valid. (CLOUD-3141)
- The S3 protocol in this release of Gateway has over 30 internal changes, in areas such as ACL, CORS, V4 signatures, and multipart writes to maintain parity with the evolution of AWS S3. (CLOUD-3142)

Fixes

- Resolved an issue that can trigger a Field Data Circuit Breaker error in Elasticsearch. (6.3.1: CLOUD-3172)
- A valid COPY request on an alias (unnamed mutable) object returned 404 Not Found. (6.3.1: CLOUD-3170)
- S3 signature errors occurred related to Date/x-amz-date with clients such as rclone. (6.3.1: CLOUD-3157)

Upgrade Impacts

To upgrade from a version of Gateway 6, see [Upgrading Gateway](#). If migrating from Elasticsearch 2.3.3 and are ready to upgrade from Gateway 5, see [Upgrading from Gateway 5.x](#).

Address the upgrade impacts for each of the versions since currently running version:

Impacts for 6.3

- *Version Requirements*
 - Swarm Storage 11.1 or higher
 - Elasticsearch 6.8.6 or 5.6.12 (for SwarmFS 2.4, migrate to ES 6.8.6)
 - Content UI 6.3, if used
- **V4 Signatures** – Gateway 6.3.1+ now matches AWS behavior in that V4-signed URLs expire in one week or less. If creating a signed URL with a longer expiration, it fails as 403 Forbidden. If a longer lasting URL is needed, use a V2-signed URL. (CLOUD-3157)

Impacts for 6.2

- *Version Requirements*
 - Swarm Storage 11.0 or higher
 - Content UI 6.2, if used
- Upgrade before using S3 clients such as Cyberduck.

Impacts for 6.1

- *Version Requirements*
 - Swarm Storage 10.2 or higher
 - Content UI 6.1, if used

Impacts for 6.0

- *Version Requirements*
 - RHEL/CentOS 7: Support for RHEL/CentOS 6 is deprecated; complete the transition to RHEL/CentOS 7 when [upgrading to Elasticsearch 5.6](#).
 - Swarm Storage 10.0 or higher.
 - Elasticsearch 5.6, with a 5.6 search index that is built on the new schema. *Do not upgrade Gateway until the 5.6 search index is complete.*
See [Migrating from Older Elasticsearch](#) and [Upgrading from Gateway 5.x](#).
 - ExpanDrive users: version 6.1.0 or higher. (CLOUD-2746)
- **New logging** – For Gateway system and audit logging, review the new, default `logging.yaml` file for any customizations to be implement. See [Gateway Configuration](#).
- Buckets named "_admin" are no longer accessible via Gateway. If a legacy _admin bucket from csmeter exists, remove it with a DELETE request directly to Swarm. (CLOUD-3025)

Watch Items and Known Issues

These are known operational limitations and watch items that exist for Gateway.

- When using the default RHEL/CentOS configuration of IPTABLES, traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s).
- Gateway is not compatible with the fingerprint scanner module for Linux PAM. If it is installed, remove it by running: `yum remove fingerprintd-pam`
- SCSP reading operations that request a Content-MD5 hash validation and for which there is a hash mismatch causes a storage node to be temporarily removed for the Gateway's connection pool due to the way Swarm reports a hash validation failure.
- Swarm Integrity Seal upgrades cannot be performed through Gateway. They may be done directly to the back-end Swarm cluster.
- If the HTTP cache control headers `If-Modified-Since` and `If-Unmodified-Since` are used, review the discussion of these in the [Storage SCSP Development](#).

The following are known issues in this release.

- A valid APPEND request on an alias (unnamed) object returns 404 Not Found. CLOUD-3181
- The AWS S3 SDK for C# does not properly sign S3-compatible requests with spaces in the name unless the domain contains ".s3." or ".s3-". See <https://github.com/aws/aws-sdk-net/issues/933>. CLOUD-3068
- When buckets are created, the `x-amz-storage-class` header is not preserved. CLOUD-3062
- The Gateway error "Failed reading from client" on a PUT due to "EofException: Early EOF" may occur when clients do not send the full body. This may point to a bug in the client's retry logic, such as not resetting the position marker to the beginning of the file or part. CLOUD-3010
- During new object creation as part of renaming with `?newname`, Gateway does not verify the user has permission to create the new object name (although it is highly likely, because it is a write within the same context). CLOUD-2966
- An `s3cmd` or `rclone` server-side copy request may time out on a multipart copy for >5GB objects (`s4cmd` performs it correctly).
Workaround: After verifying it is not the HTTPS proxy timing out, increase the client timeout: `set s3mcd socket_timeout = 600` in `~/.s3cfg` or use `rclone copy --timeout=10m --contimeout=2m caringo:mybucket/5gb caringo:mybucket/subfolder/`. CLOUD-2949
- Listings with `max-keys` may be shorter than expected because `CommonPrefixes` are included in the count of keys returned. CLOUD-2917
- Uploading files / photos using Panic's Transmit app on iOS fails due to a 403 Invalid Signature error. CLOUD-2886
- Usernames are case-insensitive, but listings exclude a token if the username (`myadmin`) does not match the case used when the token was created (`myAdmin`). CLOUD-2837
- Multipart PUT requests via recent Cyberduck versions fail with 403 SignatureDoesNotMatch when using AWS Signature Version 4. Install the Caringo `.cyberduckprofiles` from [Using the Cyberduck application with Content Gateway S3](#) which force V2 signatures. CLOUD-2799
- If a policy document includes a Principal that has plural "users" or "groups" instead of "user" or "group", the policy fails to take effect without warning. CLOUD-2783
- Versioning-enabled buckets with large numbers of objects may generate Gateway server.log warnings that can be safely ignored: "S3BucketRequestHandler: WARNING: problem with versioned bucket listing. Number of CommonPrefix (2000) exceeds max-size limit (1000)." CLOUD-2643
- 403 S3 V4 Signature mismatch errors may result when using Cyberduck with the "pound" proxy in front of Gateway S3. Workaround: Disable the Expect header in the Cyberduck preferences, or (recommended) use a different proxy such as [HAProxy](#). CLOUD-2628
- When Gateway cannot connect to Elasticsearch nodes, the errors may erroneously report this as being related to Storage nodes. CLOUD-2595
- Because of issues with Range and ETag header handling, video playback of .mp4 streams may not work correctly when served via the Gateway S3 port. It does work when served via the Gateway SCSP port. CLOUD-1964
- Gateway caches the Swarm version from the "Server:" response header, so after upgrading Swarm Gateway must be restarted to consistently see the new version. CLOUD-1271
- Gateway responds with a 500 (Internal Server Error) instead of 400 (Bad Request) if the size of the metadata headers sent to Swarm is too large. CLOUD-800
- The S3 bucket listing `StorageClass` response element always reports STANDARD. CLOUD-766

- If an S3 client escapes URI path characters such as "/", the Gateway audit log escapes the "%" characters used by the client as escape characters. URI audit log processing for S3 clients requires double-unescaping when this occurs. CLOUD-703

Content Gateway 6.4 Release

- [Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Known Issues](#)

Changes

This release of Content Gateway focuses on adding support for the next generation of SCSP SEND, which is foundational to future capabilities. SCSP SEND allows forcing an object to be written immediately another cluster for which a replication feed exists. (CLOUD-2068)

Upgrade Impacts

See [Upgrading Gateway](#). See [Upgrading from Gateway 5.x](#) if migrating from Elasticsearch 2.3.3 and are ready to upgrade from Gateway 5 to upgrade from a version of Gateway 6.

Address the upgrade impacts for each of the versions since the one being upgraded from:

Impacts for 6.4

- *Version Requirements*
 - Swarm Storage 11.2 or higher
 - Elasticsearch 6.8.6 or 5.6.12 (for SwarmFS 2.4, migrate to ES 6.8.6)
 - Content UI 6.3, if used
- **Enable service** – You need to enable the Gateway service manually after upgrading: `systemctl enable cloudgateway`. (CLOUD-3193)

Impacts for 6.3

- *Version Requirements*
 - Swarm Storage 11.1 or higher
 - Elasticsearch 6.8.6 or 5.6.12 (for SwarmFS 2.4, migrate to ES 6.8.6)
 - Content UI 6.3, if used
- **V4 Signatures** – Gateway 6.3.1+ now matches AWS behavior in that V4-signed URLs expire in one week or less. If creating a signed URL with a longer expiration, it fails as 403 Forbidden. If a longer lasting URL is needed, use a V2-signed URL. (CLOUD-3157)

Impacts for 6.2

- *Version Requirements*
 - Swarm Storage 11.0 or higher
 - Content UI 6.2, if used
- Upgrade before using S3 clients such as Cyberduck.

Impacts for 6.1

- *Version Requirements*

- Swarm Storage 10.2 or higher
- Content UI 6.1, if used

Impacts for 6.0

- *Version Requirements*
 - RHEL/CentOS 7: Support for RHEL/CentOS 6 is deprecated; complete the transition to RHEL/CentOS 7 when [upgrading to Elasticsearch 5.6](#).
 - Swarm Storage 10.0 or higher.
 - Elasticsearch 5.6, with a 5.6 search index that is built on the new schema. *Do not upgrade Gateway until the 5.6 search index is complete.*
See [Migrating from Older Elasticsearch](#) and [Upgrading from Gateway 5.x](#).
 - ExpanDrive users: version 6.1.0 or higher. (CLOUD-2746)
- **New logging** – For Gateway system and audit logging, review the new, default `logging.yaml` file for any customizations to be implement. See [Gateway Configuration](#).
- Buckets named "_admin" are no longer accessible via Gateway. If a legacy _admin bucket from csmeter exists, remove it with a DELETE request directly to Swarm. (CLOUD-3025)

Watch Items and Known Issues

These are known operational limitations and watch items that exist for Gateway.

- Traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s) when using the default RHEL/CentOS configuration of IPTABLES.
- Gateway is not compatible with Linux PAM modules that depend upon interactive validation operations such as OTP or biometric scanners.
- SCSP reading operations that request a Content-MD5 hash validation and for which there is a hash mismatch cause a storage node to be temporarily removed for the Gateway's connection pool due to the way Swarm reports a hash validation failure.
- Swarm Integrity Seal upgrades cannot be performed through Gateway. They may be done directly to the back-end Swarm cluster.
- Review the discussion of these in the [Storage SCSP Development](#) if the HTTP cache control headers `If-Modified-Since` and `If-Unmodified-Since` are used.

The following are known issues in this release.

- A valid APPEND request on an alias (unnamed) object returns 404 Not Found. CLOUD-3181
- The AWS S3 SDK for C# does not properly sign S3-compatible requests with spaces in the name unless the domain contains ".s3." or ".s3-". See <https://github.com/aws/aws-sdk-net/issues/933>. CLOUD-3068
- When buckets are created, the `x-amz-storage-class` header is not preserved. CLOUD-3062
- The Gateway error "Failed reading from client" on a PUT due to "EOFException: Early EOF" may occur when clients do not send the full body. This may point to a bug in the client's retry logic, such as not resetting the position marker to the beginning of the file or part. CLOUD-3010
- During new object creation as part of renaming with ?newname, Gateway does not verify the user has permission to create the new object name (although it is highly likely, because it is a write within the same context). CLOUD-2966
- An `s3cmd` or `rclone` server-side copy request may time out on a multipart copy for >5GB objects (`s4cmd` performs it correctly).
Workaround: After you verify the HTTPS proxy is not timing out, increase the client timeout: `set s3mcd socket_timeout = 600` in `~/s3cfg` or use `rclone copy --timeout=10m --contimeout=2m caringo:mybucket/5gb caringo:mybucket/subfolder/`. CLOUD-2949
- Listings with `max-keys` may be shorter than expected because `CommonPrefixes` are included in the count of keys returned. CLOUD-2917
- Uploading files / photos using Panic's Transmit app on iOS fails due to a 403 Invalid Signature error. CLOUD-2886
- Usernames are case-insensitive, but listings exclude a token if the username (`myadmin`) does not match the case used when the token was created (`myAdmin`). CLOUD-2837
- Multipart PUT requests via recent Cyberduck versions fail with 403 SignatureDoesNotMatch when using AWS Signature Version 4. Install the Caringo `.cyberduckprofiles` from [Using the Cyberduck application with Content Gateway S3](#) which force V2 signatures. CLOUD-2799

- The policy fails to take effect without warning if a policy document includes a Principal that has plural "users" or "groups" instead of "user" or "group". CLOUD-2783
- Versioning-enabled buckets with large numbers of objects may generate Gateway server.log warnings that can be safely ignored: "S3BucketRequestHandler: WARNING: problem with versioned bucket listing. Number of CommonPrefix (2000) exceeds max-size limit (1000)." CLOUD-2643
- 403 S3 V4 Signature mismatch errors may result when using Cyberduck with the "pound" proxy in front of Gateway S3. Workaround: Disable the Expect header in the Cyberduck preferences, or (recommended) use a different proxy such as [HAProxy](#). CLOUD-2628
- The errors may erroneously report this as being related to Storage nodes when Gateway cannot connect to Elasticsearch nodes. CLOUD-2595
- Because of issues with Range and ETag header handling, video playback of .mp4 streams may not work correctly when served via the Gateway S3 port. It does work when served via the Gateway SCSP port. CLOUD-1964
- Gateway caches the Swarm version from the "Server:" response header, so after upgrading Swarm you must restart Gateway to consistently see the new version. CLOUD-1271
- Gateway responds with a 500 (Internal Server Error) instead of 400 (Bad Request) if the size of the metadata headers sent to Swarm is too large. CLOUD-800
- The S3 bucket listing StorageClass response element always reports STANDARD. CLOUD-766
- The Gateway audit log escapes the "%" characters used by the client as escape characters if an S3 client escapes URI path characters such as "/". URI audit log processing for S3 clients require double-unescape when this occurs. CLOUD-703

Content Gateway 6.2 Release

- [Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Known Issues](#)

Changes

Support for Untenanted Objects

Untenanted objects are unnamed objects that are written to Swarm without specifying a domain. This release adds support for untenanted objects, so Gateway now accepts the Swarm setting `enforceTenancy=false`. Note: untenanted objects are incompatible with the Content UI. (UIC-409)

- **SCSP Proxy replacement** – Upgrade to Content Gateway if using SCSP Proxy because you have untenanted unnamed objects. Gateway 6.2.0 accepts untenanted objects, so it is a drop-in replacement for SCSP Proxy, which is now deprecated. (CLOUD-3136)
 - With Swarm set to `enforceTenancy=false`, you can access and continue creating untenanted objects with existing client applications.
 - By default, Gateway's `root.policy.json` provides full anonymous access, and the `idsys.json` is empty (no users). If you need to grant specific read/write access to untenanted objects, add PAM/LDAP users to the root `idsys.json` and edit the root `policy.json` to permission `GetObject`, `PutObject`, etc. as needed. See [Content Gateway Authentication](#).
- **Metrics for untenanted** – Point-in-time (`/current`) metrics are available for untenanted objects. By using `bytesSize/untentanted`, `bytesStored/untentanted`, and `objectsStored/untentanted`, you can determine the sum of content lengths, disk space used, and number of objects, respectively. See [Content Metering](#). (CLOUD-3093)

This release also includes these improvements:

- **Prometheus metrics** – The Content Gateway now generates Prometheus metrics for monitoring any dynamic features installed, such as Video Clipping. The metrics include counts of installed features, per-feature usage, and per-feature errors, as well as average time for those calls to complete. See [Managing Dynamic Features](#). (CLOUD-3123)
- **Storage node pool** – Gateway has new handling of its storage node pool in response to Swarm architecture changes, resulting in smoother performance for Swarm clients such as FileFly. (CLOUD-3101)
- Fixed: Gateway 6.2 resolves S3 bucket listing problems related to versioning and showing more than 1000 pseudo-directories. (CLOUD-2871)

Upgrade Impacts

To upgrade from a version of Gateway 6, see [Upgrading Gateway](#). If migrating from Elasticsearch 2.3.3 and are ready to upgrade from Gateway 5, see [Upgrading from Gateway 5.x](#).

Address the upgrade impacts for each of the versions since the one bring upgraded from:

Impacts for 6.2

- *Version Requirements*
 - Swarm Storage 11.0 or higher
 - Content UI 6.2, if used
- Upgrade before using S3 clients such as Cyberduck.

Impacts for 6.1

- *Version Requirements*
 - Swarm Storage 10.2 or higher
 - Content UI 6.1, if used

Impacts for 6.0

- *Version Requirements*
 - RHEL/CentOS 7: Support for RHEL/CentOS 6 is deprecated; complete the transition to RHEL/CentOS 7 when [upgrading to Elasticsearch 5.6](#).
 - Swarm Storage 10.0 or higher.
 - Elasticsearch 5.6, with a 5.6 search index that is built on the new schema. *Do not upgrade Gateway until the 5.6 search index is complete.* See [Migrating from Older Elasticsearch](#) and [Upgrading from Gateway 5.x](#).
 - ExpanDrive users: version 6.1.0 or higher. (CLOUD-2746)
- **New logging** – For Gateway system and audit logging, review the new, default `logging.yaml` file for any customizations to be implemented. See [Gateway Configuration](#).
- Buckets named "_admin" are no longer accessible via Gateway. If a legacy _admin bucket from csmeter exists, remove it with a DELETE request directly to Swarm. (CLOUD-3025)

Watch Items and Known Issues

These are known operational limitations and watch items that exist for Gateway.

- When using the default RHEL/CentOS configuration of IPTABLES, traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s).
- Gateway is not compatible with the fingerprint scanner module for Linux PAM. If it is installed, remove it by running: `yum remove fingerprint-pam`
- SCSP reading operations that request a Content-MD5 hash validation and for which there is a hash mismatch causes a storage node to be temporarily removed for the Gateway's connection pool due to the way Swarm reports a hash validation failure.
- Swarm Integrity Seal upgrades cannot be performed through Gateway. They may be done directly to the back-end Swarm cluster.
- If the HTTP cache control headers `If-Modified-Since` and `If-Unmodified-Since` are used, review the discussion of these in the [Storage SCSP Development](#).

The following are known issues in this release.

- The AWS S3 SDK for C# does not properly sign S3-compatible requests with spaces in the name unless the domain contains ".s3." or ".s3-". See <https://github.com/aws/aws-sdk-net/issues/933>. CLOUD-3068
- When buckets are created, the `x-amz-storage-class` header is not preserved. CLOUD-3062
- The Gateway error "Failed reading from client" on a PUT due to "EOFException: Early EOF" may occur when clients do not send the full body. This may point to a bug in the client's retry logic, such as not resetting the position marker to the beginning of the file or part. CLOUD-3010
- During new object creation as part of renaming with `?newname`, Gateway does not verify the user has permission to create the new object name (although it is highly likely, because it is a write within the same context). CLOUD-2966
- An `s3cmd` or `rclone` server-side copy request may time out on a multipart copy for >5GB objects (`s4cmd` performs it correctly). Workaround: After you verify it is not the HTTPS proxy timing out, increase the client timeout: `set s3cmd socket_timeout = 600` in `~/s3cfg` or use `rclone copy --timeout=10m --contimeout=2m caringo:mybucket/5gb caringo:mybucket/subfolder/`. CLOUD-2949
- Listings with `max-keys` may be shorter than expected because `CommonPrefixes` are included in the count of keys returned. CLOUD-2917
- Uploading files / photos using Panic's Transmit app on iOS fails due to a 403 Invalid Signature error. CLOUD-2886

- Usernames are case-insensitive, but listings exclude a token if the username (myadmin) does not match the case used when the token was created (myAdmin). CLOUD-2837
- Multipart PUT requests via recent Cyberduck versions fail with 403 SignatureDoesNotMatch when using AWS Signature Version 4. Install the Caringo .cyberduckprofiles from [Using the Cyberduck application with Content Gateway S3](#) which force V2 signatures. CLOUD-2799
- If a policy document includes a Principal that has plural "users" or "groups" instead of "user" or "group", the policy fails to take effect without warning. CLOUD-2783
- Versioning-enabled buckets with large numbers of objects may generate Gateway server.log warnings that can be safely ignored: "S3BucketRequestHandler: WARNING: problem with versioned bucket listing. Number of CommonPrefix (2000) exceeds max-size limit (1000)." CLOUD-2643
- 403 S3 V4 Signature mismatch errors may result when using Cyberduck with the "pound" proxy in front of Gateway S3. Workaround: Disable the Expect header in the Cyberduck preferences, or (recommended) use a different proxy such as [HAProxy](#). CLOUD-2628
- When Gateway cannot connect to Elasticsearch nodes, the errors may erroneously report this as being related to Storage nodes. CLOUD-2595
- Because of issues with Range and ETag header handling, video playback of .mp4 streams may not work correctly when served via the Gateway S3 port. It does work when served via the Gateway SCSP port. CLOUD-1964
- Gateway caches the Swarm version from the "Server:" response header, so after upgrading Swarm you must restart Gateway to consistently see the new version. CLOUD-1271
- Gateway responds with a 500 (Internal Server Error) instead of 400 (Bad Request) if the size of the metadata headers sent to Swarm is too large. CLOUD-800
- The S3 bucket listing StorageClass response element always reports STANDARD. CLOUD-766
- If an S3 client escapes URI path characters such as "/", the Gateway audit log escapes the "%" characters used by the client as escape characters. URI audit log processing for S3 clients requires double-unescaping when this occurs. CLOUD-703

Content Gateway 6.0 Release

- [Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Known Issues](#)

Changes

This release of Content Gateway includes these improvements:

- **Support for Elasticsearch 5.6** – This release of Content Gateway now completes support for Elasticsearch 5.6, so the migration away from Elasticsearch 2.3.3 can be finished and reclaim those resources. With Gateway 6.0, the newly indexed Elasticsearch 5.6 feed is made the primary search feed, which includes Swarm's new `atime` ([access time](#)) metadata for tracking content usage, if enabled. See [Migrating from Older Elasticsearch](#).
- **Upgraded logging format** – As part of support for Elasticsearch 5.6, the logging system for Content Gateway has been upgraded to log4j2, which offers more flexibility and hierarchical control. The configuration file, which was `logging.cfg`, is now `logging.yaml`. See [Gateway Configuration](#). (CLOUD-3070)
- **Graceful reboots** – Encryption keys are now persisted so rebooting of the Gateway no longer disrupts active Swarm UI browser sessions. (CLOUD-3027)
- **Health reporting** – The cluster health report providing proactive support from Swarm now includes information about the Gateway installation, including the version and configured components. This feature requires valid entries for `managementUser` and `managementPassword` in the `[storage_cluster]` configuration section. See [Gateway Configuration](#). (CLOUD-2753)

This release includes these fixes:

- The gateway error "Unable to create phone home data" was erroneously logged at startup. (CLOUD-3051)
- A rare race condition can result in a duplicate domain being created when enabling versioning in the Content UI. (CLOUD-3030)
- The service may not automatically start after a system reboot after upgrading. (CLOUD-2819)
- The `cloudgateway_server.log` had invalid SCSP warnings reporting 'Failed requests will not be retried' and 'Failed querying cluster for name and version'. (CLOUD-2663)
- S3 Gateway reports it as a failure to find the bucket when an EC write fails because of too few nodes to erasure-code a large object. (CLOUD-1452)

Upgrade Impacts

Impacts for 6.0

- *Version Requirements*
 - RHEL/CentOS 7: Support for RHEL/CentOS 6 is deprecated; complete the transition to RHEL/CentOS 7 when [upgrading to Elasticsearch 5.6](#).
 - Swarm Storage 10.0 or higher.
 - Elasticsearch 5.6, with a 5.6 search index that is built on the new schema. *Do not upgrade Gateway until the 5.6 search index is complete.* See [Migrating from Older Elasticsearch](#) and [Upgrading from Gateway 5.x](#).
 - Expandrive users: version 6.1.0 or higher. (CLOUD-2746)
- **New logging** – For Gateway system and audit logging, review the new, default `logging.yaml` file for any customizations to be implement. See [Gateway Configuration](#).

- Buckets named "_admin" are no longer accessible via Gateway. If a legacy _admin bucket from csmeter exists, remove it with a DELETE request directly to Swarm. (CLOUD-3025)

Watch Items and Known Issues

These are known operational limitations and watch items that exist for Gateway.

- Traffic to the Gateway is blocked unless action is taken to disable IPTABLES or to enable inbound traffic to the front-end protocol port(s) when using the default RHEL/CentOS configuration of IPTABLES.
- Gateway is not compatible with the fingerprint scanner module for Linux PAM. Remove it by running the following if it is installed: `yum remove fprintd-pam`
- SCSP reading operations that request a Content-MD5 hash validation and for which there is a hash mismatch causing a storage node to be temporarily removed for the Gateway's connection pool due to the way Swarm reports a hash validation failure.
- Swarm Integrity Seal upgrades cannot be performed through Gateway. They may be done directly to the back-end Swarm cluster.
- If the HTTP cache control headers `If-Modified-Since` and `If-Unmodified-Since` are used, review the discussion of these in the [Storage SCSP Development](#).

The following are known issues in this release.

- Quota states may not be properly evaluated at all times. (CLOUD-3079)
- When buckets are created, the `x-amz-storage-class` header is not preserved. (CLOUD-3062)
- The Gateway error "Failed reading from client" on a PUT due to "EofException: Early EOF" may occur when clients do not send the full body. This may point to a bug in the client's retry logic, such as not resetting the position marker to the beginning of the file or part. CLOUD-3010
- During new object creation as part of renaming with ?newname, Gateway does not verify the user has permission to create the new object name (although it is highly likely, because it is a write within the same context). CLOUD-2966
- An `s3cmd` or `rclone` server-side copy request may time out on a multipart copy for >5GB objects (`s4cmd` performs it correctly). Workaround: Increase the client timeout after verifying it is not the HTTPS proxy timing out: `set s3cmd socket_timeout = 600` in `~/s3cfg` or use `rclone copy --timeout=10m --contimeout=2m caringo:mybucket/5gb caringo:mybucket/subfolder/`. CLOUD-2949
- Listings with `max-keys` may be shorter than expected because `CommonPrefixes` are included in the count of keys returned. CLOUD-2917
- Uploading files / photos using Panic's Transmit app on iOS fails due to a 403 Invalid Signature error. CLOUD-2886
- Gateway 5.2.2 and earlier do not output the `NextMarker` field in S3 listings, which can cause some S3 clients such as Caringo Drive, `rclone`, and Transmit to show only 1000 files in a directory or to miss some subdirectories. CLOUD-2871
- Usernames are case-insensitive, but listings exclude a token if the username (`myadmin`) does not match the case used when the token was created (`myAdmin`). CLOUD-2837
- Multipart PUT requests via recent Cyberduck versions fail with 403 SignatureDoesNotMatch when using AWS Signature Version 4. Install the Caringo `.cyberduckprofiles` from [Using the Cyberduck application with Content Gateway S3](#) which force V2 signatures. CLOUD-2799
- The policy fails to take effect without warning if a policy document includes a `Principal` that has plural "users" or "groups" instead of "user" or "group", . CLOUD-2783
- Versioning-enabled buckets with large numbers of objects may generate Gateway `server.log` warnings that can be safely ignored: "S3BucketRequestHandler: WARNING: problem with versioned bucket listing. Number of CommonPrefix (2000) exceeds max-size limit (1000)." CLOUD-2643
- 403 S3 V4 Signature mismatch errors may result when using Cyberduck with the "pound" proxy in front of Gateway S3. Workaround: Disable the Expect header in the Cyberduck preferences, or (recommended) use a different proxy such as [HAProxy](#). CLOUD-2628
- When Gateway cannot connect to Elasticsearch nodes, the errors may erroneously report this as being related to Storage nodes. CLOUD-2595
- Because of issues with Range and ETag header handling, video playback of .mp4 streams may not work correctly when served via the Gateway S3 port. It does work when served via the Gateway SCSP port. CLOUD-1964
- Gateway caches the Swarm version from the "Server:" response header, so after upgrading Swarm Gateway must be restarted to consistently see the new version. CLOUD-1271
- Gateway responds with a 500 (Internal Server Error) instead of 400 (Bad Request) if the size of the metadata headers sent to Swarm is too large. CLOUD-800

- The S3 bucket listing StorageClass response element always reports STANDARD. CLOUD-766
- The Gateway audit log escapes the "%" characters used by the client as escape characters if an S3 client escapes URI path characters such as "/". URI audit log processing for S3 clients require double-unescaping when this occurs. CLOUD-703

Content UI Release Notes

The Swarm Content UI is Gateway's cloud interface to Swarm-based content, providing end users with direct browser access to content. The Content UI simplifies content management (such as configuring tenants and storage domains and creating search collections based on custom metadata tags), and it also enables end users to download and upload content directly to and from file systems.

- [Content UI 7.6 Release](#)
- [Content UI 7.5 Release](#)
- [Content UI 7 Release](#)
- [Content UI 6 Release](#)

Content UI 7.6 Release

- [Changes in Content UI 7.6](#)
- [Changes in Content UI 7.5](#)
- [Changes in Content UI 7.4](#)
- [Changes in Content UI 7.3](#)
- [Changes in Content UI 7.2](#)
- [Changes in Content UI 7.1](#)
- [Changes in Content UI 7.0](#)

Changes in Content UI 7.6

- Added support for [Bucket Lifecycle Policy](#) with S3 and SCSP.

Upgrade Impacts

Version Requirements:

- Gateway 7.9

Watch Items and Issues

- When attempting to delete all versions of an object with locking applied to at least one version, the operation fails. (UIC-577)
- When an object is locked, the delete operation may still be available on that object. This behavior mirrors S3. The protected version(s) are still present, protected, and accessible in the storage cluster, but the object is longer be visible in Content UI. This happens due to the delete marker available in the "current version". To make the object visible again, either use an S3 tool to remove the delete marker or upload a new version of the object. (UIC-576)
- *Inherit Versioning* in domain or bucket properties is not working as expected. To make the versioning work, both domain and bucket settings must be "Enabled" and Swarm settings *policy.versioning* must be "allowed". (UIC-272)

Changes in Content UI 7.5

- Added support for [Hybrid Cloud Copy to S3](#).
- Added support for ransomware protection using [object locking](#). This includes both configuring bucket-level locking defaults as well as fine-grained object version locking support.

Upgrade Impacts

Version Requirements:

- Gateway 7.8

Watch Items and Issues

- When attempting to delete all versions of an object with locking applied to at least one version, the operation fails. (UIC-577)

- When an object is locked, the delete operation may still be available on that object. This behavior mirrors S3. The protected version(s) are still present, protected, and accessible in the storage cluster, but the object is longer be visible in Content UI. This happens due to the delete marker available in the "current version". To make the object visible again, either use an S3 tool to remove the delete marker or upload a new version of the object. (UIC-576)
- *Inherit Versioning* in domain or bucket properties is not working as expected. To make the versioning work, both domain and bucket settings must be "Enabled" and Swarm settings *policy.versioning* must be "allowed". (UIC-272)
- For tenant users to be able to navigate to the own domain, grant them *ListDomains* permission in the tenant policy. Content Portal is missing that action, but it can be added in the policy via JSON text editor. See <https://caringo.atlassian.net/wiki/pages/createpage.action?spaceKey=DOCS&title=Content%20UI%20Installation>. (UIC-580)

Changes in Content UI 7.4

- Added drag-and-drop functionality within the folder listing views
- Added finer-grained control for data protection policies
- Third-party software package updates

Upgrade Impacts

Version requirements:

- Gateway 7.4

Changes in Content UI 7.3

This release contains support for the System domain, allowing you to use Swarm's modern features such as metadata searching and policy/access control for unnamed and untenanted objects through the UI. See [System Domain for Legacy Objects](#). (UIC-479, UIC-445)

Watch Items and Issues

- Removes operational limitation of using single sign-on with tenant-level IDSYS or logins with *user+tenant* style user names. (CLOUD-3229)
- Remaining items from 7.0 are unchanged.

Changes in Content UI 7.2

This release contains improvements to SAML behaviors and token handling for tenants. (UIC-454, UIC-453, UIC-439)

Watch Items and Issues – Same as 7.0.

Changes in Content UI 7.1

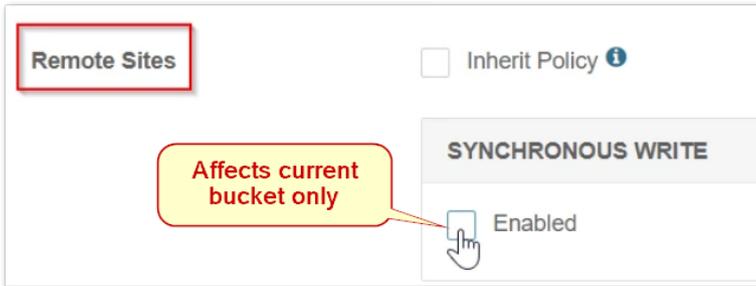
Updating to this version is recommended to resolve potential issues with the setup of Remote Synchronous Write and with moving between the Content and Storage UIs. (UIC-441, UIC-443)

Watch Items and Issues – Same as 7.0.

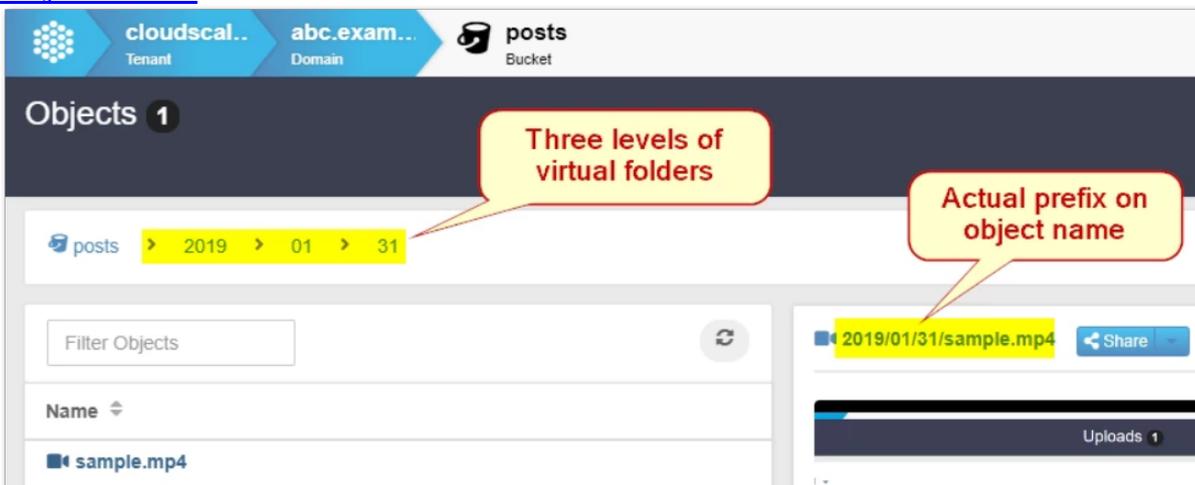
Changes in Content UI 7.0

Remote Synchronous Write – Specific domains and buckets can be configured to broadcast the new content to all remote sites immediately in the Content UI. This feature, *Remote Synchronous Write*, delays write completion until replicas exist in every remote cluster. This setting allows you support applications that require guarantees that backups are committed to every site, and to support publishing requirements to be able to read new content from any remote site immediately after ingesting.

See [Setting Remote Synchronous Write \(RSW\)](#).



Folder Listing UI – With Gateway 7, folder listing support across Swarm clients (Content UI, SwarmFS, and S3) has been rearchitected and centralized. *Folder listing* is what renders the virtual folders (prefixes) on named Swarm objects (such as `FY2019/Q3/object.jpg`) in to familiar folders on users' file systems. The service makes full use of newer Elasticsearch features and is no longer bound by ES listing limits. See [Using Virtual Folders](#).



The Content UI now presents these folders as walkable directories, with these key benefits:

- **Prefix filtering** – The Content UI provides users a fast and intuitive way to view and manage content in the bucket automatically by parsing object prefixes into hierarchical folders in real-time.
- **Empty folders** – The Content UI allows *creation* and persistence of new, empty folders ready to receive files. This allows you to plan and set up organizing structures ahead of time, to guide content uploaders to use your organization. By having users upload directly to your folders, you can enforce a content architecture and avoid the risk they perform bulk uploads using a malformed prefix.
- **Recursive deletes** – The folder listing feature of Content UI includes the convenience of recursive delete (deleting the folder, contents, as well as any subfolders and contents). The created folder can be deleted from the flattened view of the folder listing, without disturbing contents or the prefix naming. (UIC-161)

Single Sign-on – The Content UI can now offer SSO (single sign-on) for your users through the new SAML 2.0 support in Content Gateway. The login page detects any SAML configuration for the requested host or domain and redirects the user to log in with your identity providers, such as OneLogin, Okta, or Google. You can implement a single sign-on at the root level and/or for specific domains. See [Enabling SSO with SAML](#). (UIC-212)



Watch Items and Issues

These are current operational limitations:

- Single sign-on does not support tenant-level IDSYS or logins with *user+tenant* style user names. (CLOUD-3229)
- After upgrading the Content UI (Portal) and/or Storage UI, the cache must be cleared to get the new version. Either shift-Reload the page or clear the browser cache, then verify the **About** page shows the new version. (UIC-222)

These are known issues:

- Under the Permissions tab, the Access Policy wizard offers Actions related to Tokens, but these are incompletely implemented. Denying users the ability to create tokens prevents them from accessing the Content UI. (UIC-406)
- When uploading files from iOS devices, all filenames are "image.jpg". These may be uploaded as UUIDs or upload one file at a time with a different filename prefix. (UIC-188)
- The login button for the Content UI may require a second click to proceed. (UIC-98)
- New tokens with a user-supplied description do not initially show the description in the list of tokens. Workaround: Refresh the list of tokens after adding a new token. (UIC-43)
- To create a collection from metadata that includes non-ASCII characters, you must create it from the domain page. The ability to create a collection from non-ASCII metadata on the object details page is not currently supported. (UIC-31)
- For any collections that erroneously show the bucket icon, make a small edit and resave them, which causes them to be fully repaired. (UIC-24)

Content UI 7.5 Release

- [Changes in Content UI 7.5](#)
- [Changes in Content UI 7.4](#)
- [Changes in Content UI 7.3](#)
- [Changes in Content UI 7.2](#)
- [Changes in Content UI 7.1](#)
- [Changes in Content UI 7.0](#)

Changes in Content UI 7.5

- Added support for [Hybrid Cloud Copy to S3](#).
- Added support for ransomware protection using [object locking](#). This includes both configuring bucket-level locking defaults as well as fine-grained object version locking support.

Upgrade Impacts

Version Requirements:

- Gateway 7.8

Watch Items and Issues

- When attempting to delete all versions of an object with locking applied to at least one version, the operation fails. (UIC-577)
- When an object is locked, the delete operation may still be available on that object. This behavior mirrors S3. The protected version(s) are still present, protected, and accessible in the storage cluster, but the object is longer be visible in Content UI. This happens due to the delete marker available in the "current version". To make the object visible again, either use an S3 tool to remove the delete marker or upload a new version of the object. (UIC-576)
- *Inherit Versioning* in domain or bucket properties is not working as expected. To make the versioning work, both domain and bucket settings must be "Enabled" and Swarm settings *policy.versioning* must be "allowed". (UIC-272)
- For tenant users to be able to navigate to the own domain, grant them *ListDomains* permission in the tenant policy. Content Portal is missing that action, but it can be added in the policy via JSON text editor. See [Content UI Installation](#). (UIC-580)

Changes in Content UI 7.4

- Added drag-and-drop functionality within the folder listing views
- Added finer-grained control for data protection policies
- Third-party software package updates

Upgrade Impacts

Version requirements:

- Gateway 7.4

Changes in Content UI 7.3

This release contains support for the System domain, allowing you to use Swarm's modern features such as metadata searching and policy/access control for unnamed and untenanted objects through the UI. See [System Domain for Legacy Objects](#). (UIC-479, UIC-445)

Watch Items and Issues

- Removes operational limitation of using single sign-on with tenant-level IDSYS or logins with *user+tenant* style user names. (CLOUD-3229)
- Remaining items from 7.0 are unchanged.

Changes in Content UI 7.2

This release contains improvements to SAML behaviors and token handling for tenants. (UIC-454, UIC-453, UIC-439)

Watch Items and Issues – Same as 7.0.

Changes in Content UI 7.1

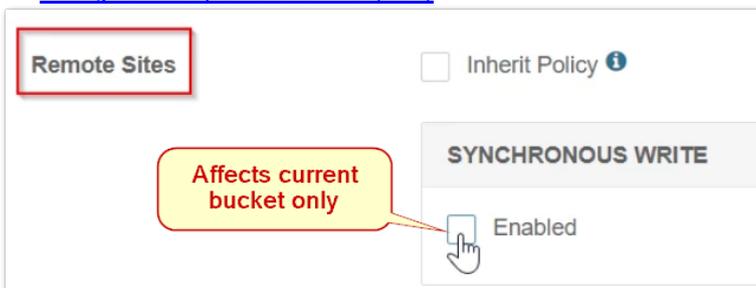
Updating to this version is recommended to resolve potential issues with the setup of Remote Synchronous Write and with moving between the Content and Storage UIs. (UIC-441, UIC-443)

Watch Items and Issues – Same as 7.0.

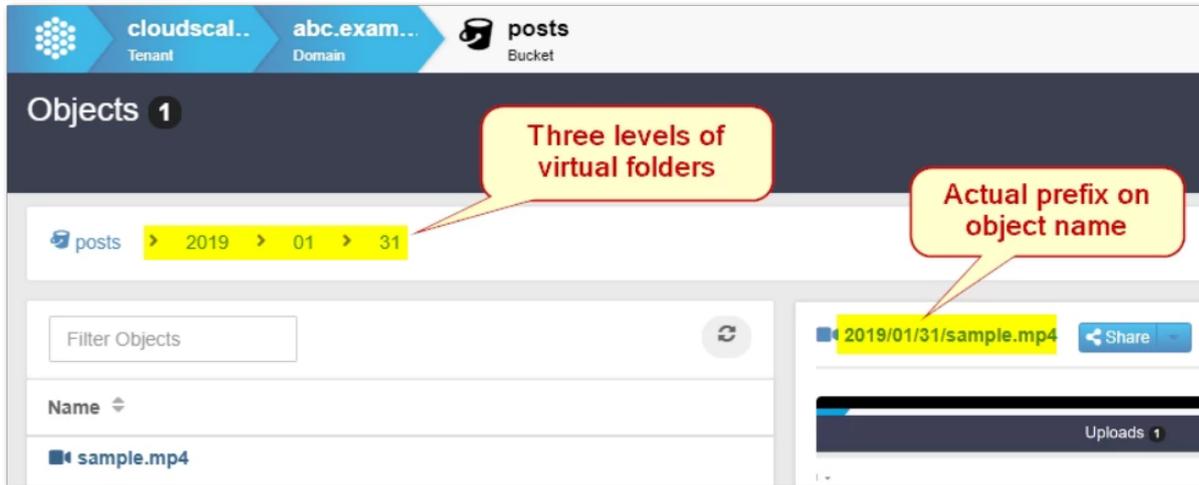
Changes in Content UI 7.0

Remote Synchronous Write – Specific domains and buckets can be configured to broadcast the new content to all remote sites immediately in the Content UI. This feature, *Remote Synchronous Write*, delays write completion until replicas exist in every remote cluster. This setting allows you support applications that require guarantees that backups are committed to every site, and to support publishing requirements to be able to read new content from any remote site immediately after ingesting.

See [Setting Remote Synchronous Write \(RSW\)](#).



Folder Listing UI – With Gateway 7, folder listing support across Swarm clients (Content UI, SwarmFS, and S3) has been rearchitected and centralized. *Folder listing* is what renders the virtual folders (prefixes) on named Swarm objects (such as `2019/Q3/object.jpg`) in to familiar folders on users' file systems. The service makes full use of newer Elasticsearch features and is no longer bound by ES listing limits. See [Using Virtual Folders](#).



The Content UI now presents these folders as walkable directories, with these key benefits:

- **Prefix filtering** – The Content UI provides users a fast and intuitive way to view and manage content in the bucket automatically by parsing object prefixes into hierarchical folders in real-time.
- **Empty folders** – The Content UI allows *creation* and persistence of new, empty folders ready to receive files. This allows you to plan and set up organizing structures ahead of time, to guide content uploaders to use your organization. By having users upload directly to your folders, you can enforce a content architecture and avoid the risk they perform bulk uploads using a malformed prefix.
- **Recursive deletes** – The folder listing feature of Content UI includes the convenience of recursive delete (deleting the folder, contents, as well as any subfolders and contents). The created folder can be deleted from the flattened view of the folder listing, without disturbing contents or the prefix naming. (UIC-161)

Single Sign-on – The Content UI can now offer SSO (single sign-on) for your users through the new SAML 2.0 support in Content Gateway. The login page detects any SAML configuration for the requested host or domain and redirects the user to log in with your identity providers, such as OneLogin, Okta, or Google. You can implement a single sign-on at the root level and/or for specific domains. See [Enabling SSO with SAML](#). (UIC-212)



Watch Items and Issues

These are current operational limitations:

- Single sign-on does not support tenant-level IDSYS or logins with *user+tenant* style user names. (CLOUD-3229)
- After upgrading the Content UI (Portal) and/or Storage UI, the cache must be cleared to get the new version. Either shift-Reload the page or clear the browser cache, then verify the **About** page shows the new version. (UIC-222)

These are known issues:

- Under the Permissions tab, the Access Policy wizard offers Actions related to Tokens, but these are incompletely implemented. Denying users the ability to create tokens prevents them from accessing the Content UI. (UIC-406)
- When uploading files from iOS devices, all filenames are "image.jpg". These may be uploaded as UUIDs or upload one file at a time with a different filename prefix. (UIC-188)
- The login button for the Content UI may require a second click to proceed. (UIC-98)

- New tokens with a user-supplied description do not initially show the description in the list of tokens. Workaround: Refresh the list of tokens after adding a new token. (UIC-43)
- To create a collection from metadata that includes non-ASCII characters, you must create it from the domain page. The ability to create a collection from non-ASCII metadata on the object details page is not currently supported. (UIC-31)
- For any collections that erroneously show the bucket icon, make a small edit and resave them, which causes them to be fully repaired. (UIC-24)

Content UI 7 Release

- [Changes in Content UI 7.4](#)
- [Changes in Content UI 7.3](#)
- [Changes in Content UI 7.2](#)
- [Changes in Content UI 7.1](#)
- [Changes in Content UI 7.0](#)

Changes in Content UI 7.4

- Added drag-and-drop functionality within the folder listing views
- Added finer-grained control for data protection policies
- Third party software package updates

Upgrade Impacts

Version requirements:

- Gateway 7.4

Changes in Content UI 7.3

This release contains support for System domain, allowing use of Swarm's modern features such as metadata searching and policy/access control for unnamed and untenanted objects through the UI. See [System Domain for Legacy Objects](#). (UIC-479, UIC-445)

Watch Items and Issues

- Removes operational limitation of using single sign-on with tenant-level IDSYS or logins with *user+tenant* style user names. (CLOUD-3229)
- Remaining items from 7.0 unchanged.

Changes in Content UI 7.2

This release contains improvements to SAML behaviors and token handling for tenants. (UIC-454, UIC-453, UIC-439)

Watch Items and Issues – Same as 7.0.

Changes in Content UI 7.1

Updating to this version is recommended to resolve potential issues with the setup of Remote Synchronous Write and with moving between the Content and Storage UIs. (UIC-441, UIC-443)

Watch Items and Issues – Same as 7.0.

Changes in Content UI 7.0

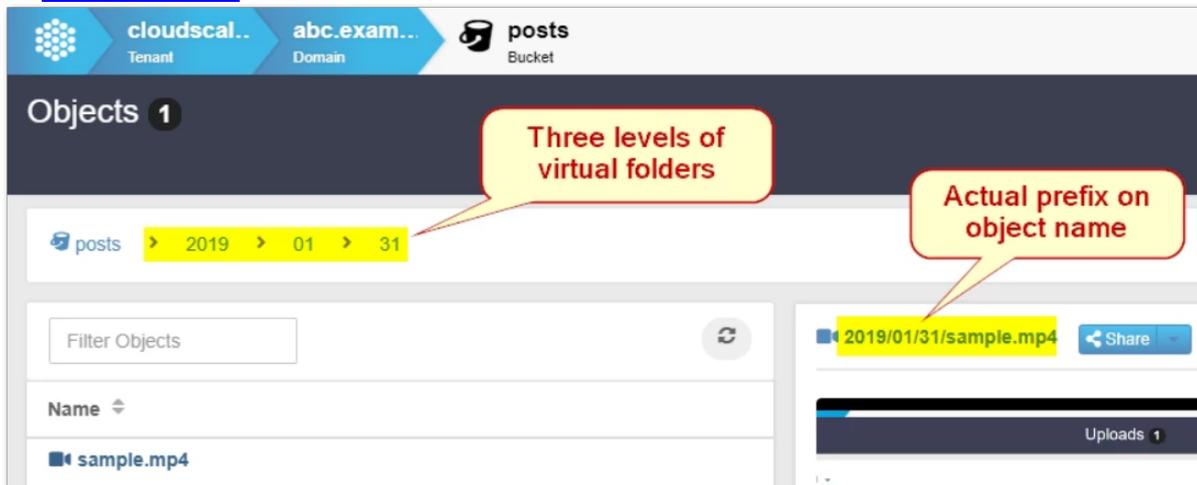
Remote Synchronous Write – In the Content UI, configure specific domains and buckets to broadcast new content to all remote sites immediately. This feature, *Remote Synchronous Write*, delays write completion until replicas exist in every remote cluster. This setting allows support applications requiring guarantees backups are committed to every site, and to support publishing requirements to be able to read new content from any remote site immediately after ingest.

See [Setting Remote Synchronous Write \(RSW\)](#).



Folder Listing UI – With Gateway 7, folder listing support across Swarm clients (Content UI, SwarmFS, and S3) has been rearchitected and centralized. *Folder listing* is what renders the virtual folders (prefixes) on named Swarm objects (such as `FY2019/Q3/object.jpg`) in to familiar folders on the users' file systems. The service makes full use of newer Elasticsearch features and is no longer bound by ES listing limits.

See [Using Virtual Folders](#).



The Content UI now presents these folders as walkable directories, with these key benefits:

- **Prefix filtering** – The Content UI provides users a fast and intuitive way to automatically view and manage content in a bucket by parsing object prefixes in to hierarchical folders in real time.
- **Empty folders** – The Content UI allows *creation* and persistence of new, empty folders ready to receive files. This allows planning and setting up organizing structures ahead of time, to guide content uploaders to use the organization. A content architecture can be enforced and avoid the risk users perform bulk uploads using a malformed prefix by having them upload directly to folders.
- **Recursive deletes** – The folder listing feature of Content UI includes the convenience of recursive delete (deleting the folder, contents, as well as any subfolders and contents). From the flattened view of the folder listing, the created folder can be deleted without disturbing the contents or the prefix naming. (UIC-161)

Single Sign-on – The Content UI can now offer SSO (single sign-on) for users through the new SAML 2.0 support in Content Gateway. The login page detects any SAML configuration for the requested host or domain and redirects the user to log in with the identity provider, such as OneLogin, Okta, or Google. Single sign-on can be implemented at the root level and/or for specific domains. See [Enabling SSO with SAML](#). (UIC-212)



Watch Items and Issues

These are current operational limitations:

- Single sign-on does not support tenant-level IDSYS or logins with *user+tenant* style user names. (CLOUD-3229)
- The cache must be cleared to get the new version after upgrading the Content UI (Portal) and/or Storage UI. Either shift-Reload the page or clear the browser cache, then verify the **About** page shows the new version. (UIC-222)

These are known issues:

- Under the Permissions tab, the Access Policy wizard offers Actions related to Tokens, but these are incompletely implemented. Denying users the ability to create tokens prevents them from accessing the Content UI. (UIC-406)
- All filenames are "image.jpg" when uploading files from iOS devices. These may be uploaded as UUIDs or uploaded one file at a time with a different filename prefix. (UIC-188)
- The login button for the Content UI may require a second click to proceed. (UIC-98)
- New tokens with user-supplied description do not initially show the description in the list of tokens. Workaround: Refresh the list of tokens after adding a new token. (UIC-43)
- Create a collection from the domain page to create it from metadata including non-ASCII characters. The ability to create a collection from non-ASCII metadata on the object details page is not currently supported. (UIC-31)
- Make a small edit and resave a collection for any erroneously showing the bucket icon, which causes them to be fully repaired. (UIC-24)

Content UI 6 Release

- [Changes in Content UI 6.3](#)
- [Changes in Content UI 6.2](#)
- [Changes in Content UI 6.1](#)
- [Changes in Content UI 6.0](#)

Changes in Content UI 6.3

This release includes an improvement to uploading large files:

- Improved: The part size for multipart uploads is increased to 25 MB, a common S3 client default part size. This change speeds the performance of large uploads and avoids error responses. (UIC-407)
- Fixed: After a video clip is successfully created, clicking the "new clip will be available here" link erroneously results in a red error banner. Clicking the link a second time resolves to the clip. (6.3.1: UIC-421)
- Fixed: Adding a collection does not work in version 6.2. (UIC-411)

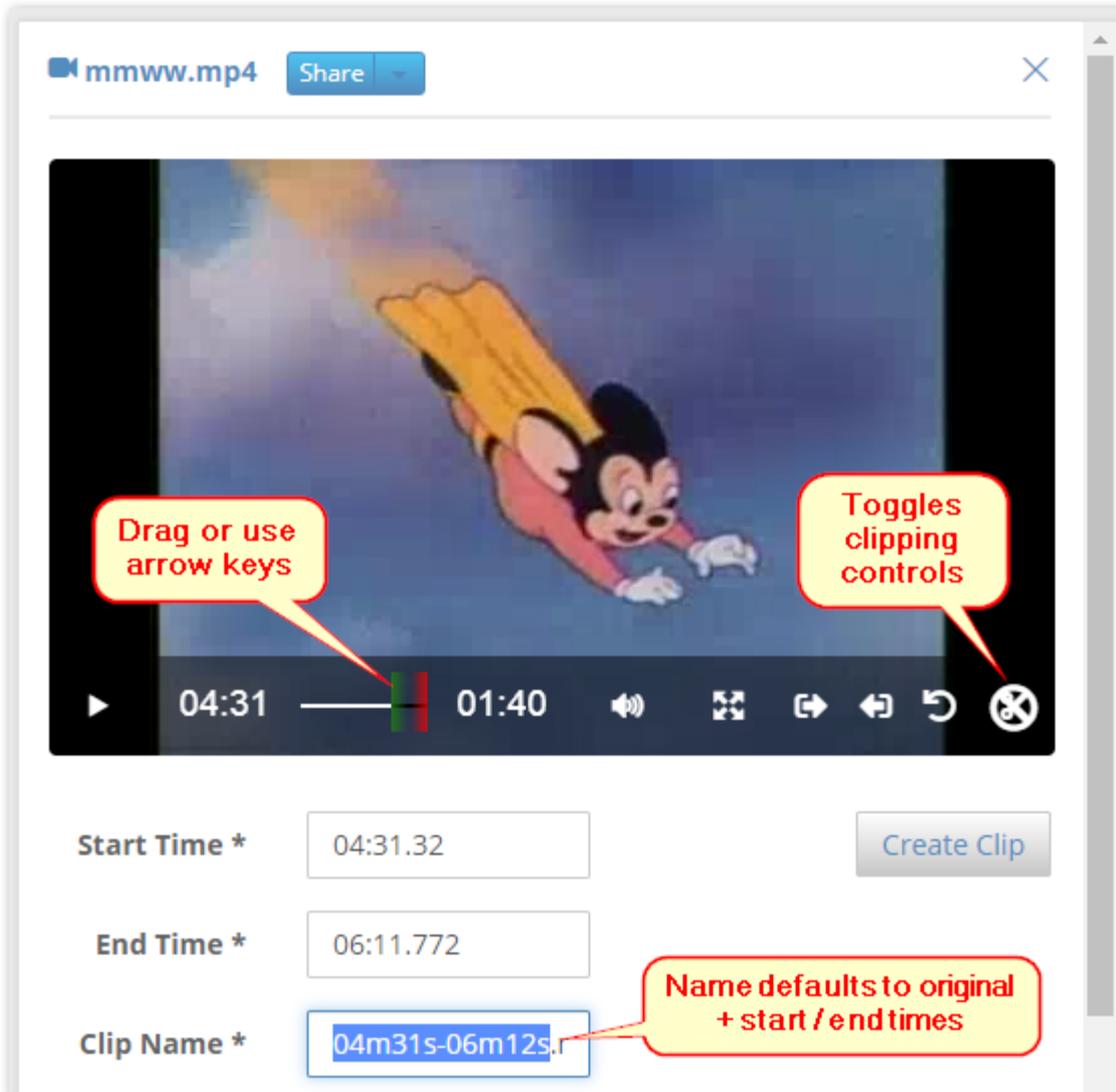
Known Issues

- Under the Permissions tab, the Access Policy wizard offers Actions related to Tokens, but these are incompletely implemented. Denying users the ability to create tokens prevents them from accessing the Content UI. (UIC-406)
- After upgrading the Content UI (Portal) and/or Storage UI, the cache must be cleared to get the new version. Either shift-Reload the page or clear the browser cache, then verify the About page shows the new version. A logout does not fix it nor is it necessary. (UIC-222)
- When uploading files from iOS devices, all filenames are "image.jpg". You may upload these as UUIDs or upload one file at a time with a different filename prefix. (UIC-188)
- The login button for the Content UI may require a second click to proceed. (UIC-98)
- New tokens with a user-supplied description do not initially show the description in the list of tokens. Workaround: Refresh the list of tokens after adding a new token. (UIC-43)
- To create a collection from metadata that includes non-ASCII characters, you must create it from the domain page. The ability to create a collection from non-ASCII metadata on the object details page is not currently supported. (UIC-31)
- For any collections that erroneously show the bucket icon, make a small edit and resave them, which causes them to be fully repaired. (UIC-24)

Changes in Content UI 6.2

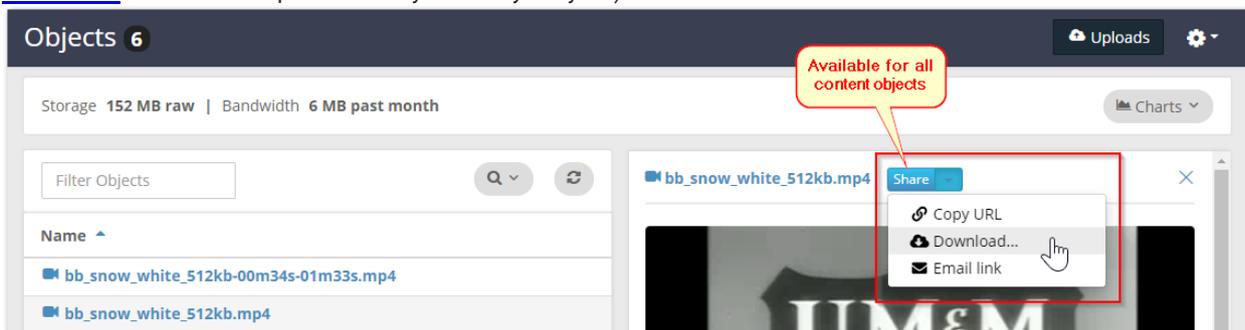
This release includes major functional enhancements, including video-clipping, content sharing, and expanded bulk upload support.

- **Video clipping for Partial File Restore** – The Content UI now supports creation of video clips (part of the functionality known as *partial file restore*) from videos stored in Swarm. This occurs directly through the browser interface, with no downloads/uploads or local editing tools needed. When this optional feature is installed in Content Gateway, videos viewed in the UI show a scissor icon to the right of the playback controls, which toggles the video clip creation tools. Each clip is a standalone video with a start and end time relative to the source video, and it is saved to a new name. Resulting clips have no dependency on the source files. Supported HTML5 video formats are MPEG-4/MP4 (H.264) and WebM. (UIC-353) See [Video Clipping for Partial File Restore](#).



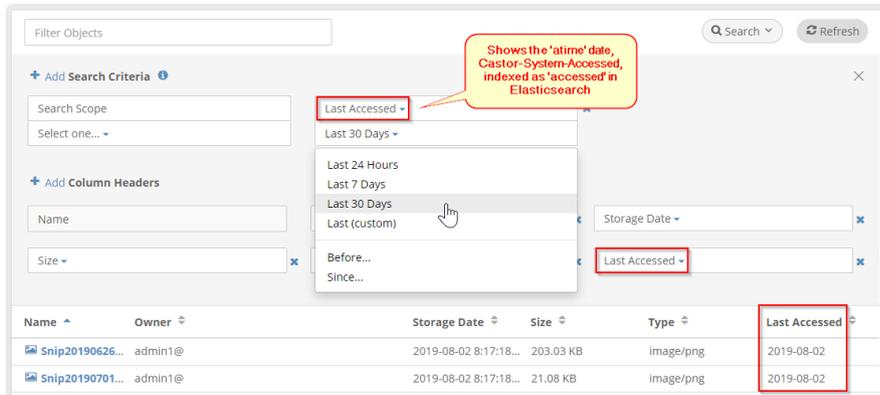
- Sharing controls on all content** – When you select an object in a listing to view it, a new **Share** button appears next to its name. The button opens a menu of commands for content sharing, including copying the URL to your local clipboard, downloading the file locally, and opening the default email program to email the link to someone else. Note: the object remains protected by the bucket's permissions; if you regularly want to share links to objects

that are too large for email with others *outside* of your organization, consider creating a public bucket for that purpose (see [Setting Permissions](#) and use the template *Read-Only Access by Everyone*).



- Large uploads through Content UI** – The Content UI file uploader is redesigned to write directly to Swarm storage and bypass spooling altogether, which removes the prior 4 GB limit. Now the Content UI accepts more and larger files and is better able to recover and resume uploads that encounter errors. (UIC-399)

- Filtering for Time of Last Access** – When you add columns and search criteria to your object filters, you can now show and filter on **Last Accessed**, which is the optional [atime feature](#) captured in the `Castor-System-Accessed` header and indexed in Elasticsearch as 'accessed'. Filtering on the time of last access includes standard and custom time spans from the present as well as Before and Since ranges. (UIC-374)



Fixed: Content UI did not permit saving of valid erasure-coding policies that had more parity segments than data segments (such as 5:7). (UIC-391)

Known Issues

- Adding a collection does not work in version 6.2. (UIC-411)
- Under the Permissions tab, the Access Policy wizard offers Actions related to Tokens, but these are incompletely implemented. Denying users the ability to

create tokens prevents them from accessing the Content UI. (UIC-406)

- After upgrading the Content UI (Portal) and/or Storage UI, the cache must be cleared to get the new version. Either shift-Reload the page or clear the browser cache, then verify the About page shows the new version. A logout does not fix it nor is it necessary. (UIC-222)
- When uploading files from iOS devices, all filenames are "image.jpg". You may upload these as UUIDs or upload one file at a time with a different filename prefix. (UIC-188)
- The login button for the Content UI may require a second click to proceed. (UIC-98)
- New tokens with a user-supplied description do not initially show the description in the list of tokens. Workaround: Refresh the list of tokens after adding a new token. (UIC-43)
- To create a collection from metadata that includes non-ASCII characters, you must create it from the domain page. The ability to create a collection from non-ASCII metadata on the object details page is not currently supported. (UIC-31)
- For any collections that erroneously show the bucket icon, make a small edit and resave them, which causes them to be fully repaired. (UIC-24)

Changes in Content UI 6.1

This release corrects several issues with the display of charts in Content UI.

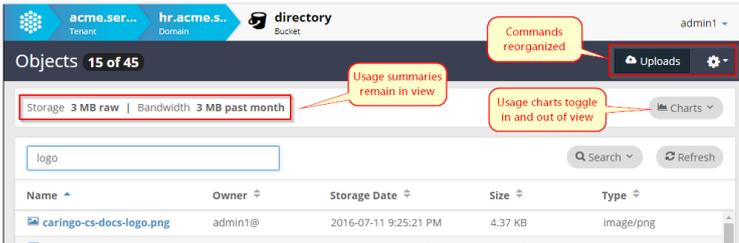
Known Issues

- After upgrading the Content UI (Portal) and/or Storage UI, the cache must be cleared to get the new version. Either shift-Reload the page or clear the browser cache, then verify the About page shows the new version. A logout does not fix it nor is it necessary. (UIC-222)
- When uploading files from iOS devices, all filenames are "image.jpg". These may be uploaded as UUIDs or uploaded one file at a time with a different filename prefix. (UIC-188)
- The login button for the Content UI may require a second click to proceed. (UIC-98)
- New tokens with a user-supplied description do not initially show the description in the list of tokens. Workaround: Refresh the list of tokens after adding a new token. (UIC-43)
- To create a collection from metadata that includes non-ASCII characters, you must create it from the domain page. The ability to create a collection from non-ASCII metadata on the object details page is not currently supported. (UIC-31)
- For any collections that erroneously show the bucket icon, make a small edit and resave them, which causes them to be fully repaired. (UIC-24)

Changes in Content UI 6.0

In support of Swarm 10, Content UI has extensive usability upgrades in response to client feedback. These are highlights:

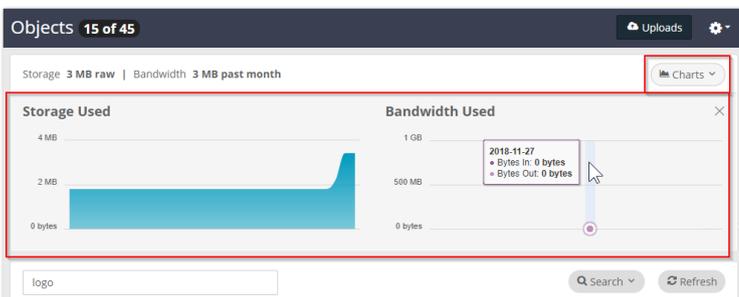
Streamlined Usage Reports – Page layout and navigation across the Content UI are reorganized to be more efficient. Storage and Bandwidth usage summaries are moved into the title bar of the Charts panel so they remain in view even when collapsed because of the importance they provide. Clicking **Charts** expands and collapses the full view of the charts:



Commands and Properties – For consistency and simplicity, the commands and properties are all unified under the gear icon:

Commands such as **Upload** appear where needed:

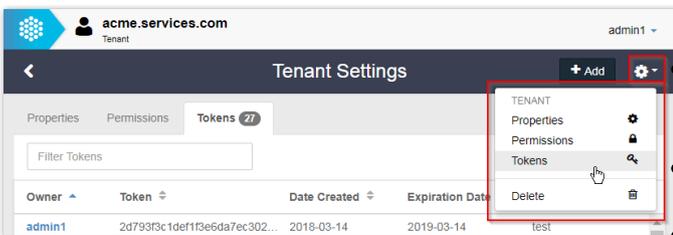
Search and Collections – Handling of search collection creation and filters is improved, with the **Filter** and **Search** functions unified in a collapsible panel expanded by clicking **Search**:



By selecting **Create Collection** on the object view, you can create a metadata-based search in a single click, which greatly speeds up the design of your search criteria:

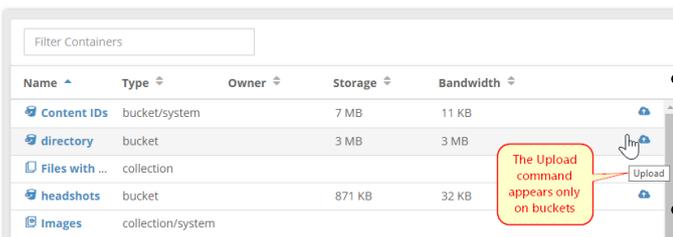
Known Issues

- After upgrading the Content UI (Portal) and/or Storage UI, the cache must be cleared to get the new version. Either shift-Reload the page or clear the browser cache, then verify the About page shows the new version. A logout does not fix it nor is it necessary. (UIC-222)



- When uploading files from iOS devices, all filenames are "image.jpg". You may upload these as UUIDs or upload one file at a time with a different filename prefix. (UIC-188)

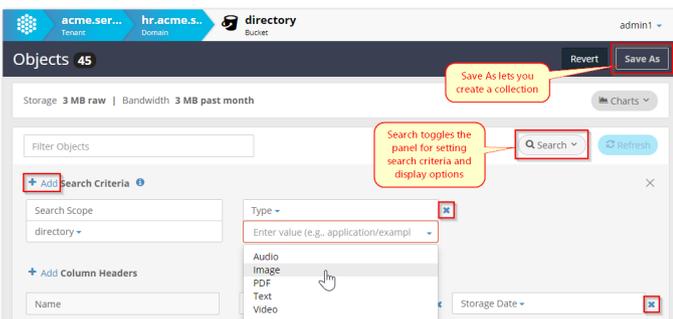
- The login button for the Content UI may require a second click to proceed. (UIC-98)

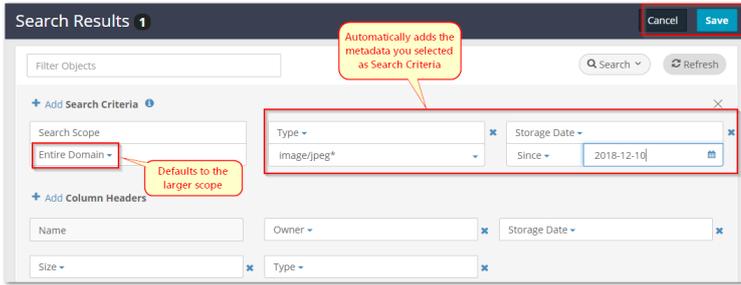


- New tokens with a user-supplied description do not initially show the description in the list of tokens. Workaround: Refresh the list of tokens after adding a new token. (UIC-43)

- To create a collection from metadata that includes non-ASCII characters, you must create it from the domain page. The ability to create a collection from non-ASCII metadata on the object details page is not currently supported. (UIC-31)

- For any collections that erroneously show the bucket icon, make a small edit and resave them, which causes them to be fully repaired. (UIC-24)





The screenshot shows the 'Search Results 1' interface. At the top right, there are 'Cancel' and 'Save' buttons. Below the title bar is a 'Filter Objects' input field, a search icon, and a 'Refresh' button. The main area is divided into three sections: 'Add Search Criteria', 'Add Column Headers', and a list of search results. The 'Add Search Criteria' section contains a 'Search Scope' dropdown set to 'Entire Domain', a 'Type' dropdown set to 'Image/jpeg*', and a 'Storage Date' dropdown set to 'Since 2018-12-10'. The 'Add Column Headers' section contains 'Name', 'Owner', and 'Storage Date' dropdowns. A callout box points to the 'Search Scope' dropdown with the text 'Defaults to the larger scope'. Another callout box points to the 'Type' and 'Storage Date' dropdowns with the text 'Automatically adds the metadata you selected as Search Criteria'.

SDK Release Notes

The Swarm Software Development Kit (SDK) simplifies integration with Swarm by providing client library support for specific Simple Content Storage Protocol (SCSP) operations. The SDK assists developers by implementing a consistent set of features using a common API in each supported programming language.

- [SDK version 9.1.0](#)
 - [Best practice](#)
- [SDK version 6.1.5](#)
- [SDK version 6.1.4](#)
- [SDK version 6.1.3](#)
- [SDK version 6.1.2](#)
- [Limitations](#)
- [Deprecation Notices](#)
- [Application and Configuration Notes](#)

SDK version 9.1.0

The SDK version was updated to reflect the version of Swarm testing and compatibility. This release includes the following enhancements and changes:

- The Java SDK is now built on Apache HttpComponents HttpClient 4.5.2 and HttpCore 4.4.4.
- The Java SDK testing was done against Java 8.
- The C++ SDK fixes a memory leak on redirect.
- The Python and C# SDKs now correctly handle a 202 response when the request is sent with `Expect: 100-continue`. This is important for multipart completion handling with large numbers of parts (and therefore a large manifest in the request body). This fix means all SDKs support Swarm multipart completion and multipart copy-by-part requests.



Best practice

Content Gateway drops trailer headers, so have your clients parse response bodies instead of relying on the trailers for completion headers.

SDK version 6.1.5

This release includes the following enhancements and changes:

- The Java SDK now includes several new classes and methods for facilitating remote replication for an object as well as synchronously writing to a local and remote cluster. Reference [SDK for Java](#) for more details.
- The delete methods of `ScspDomain` and `ScspBucket` classes of the Java SDK add `?recursive=yes` if there is no recursive query argument on the call. This allows users to pass `recursive=no` to effect immediate content deletion for all content in a context.
- The `ScspDomain.create` call no longer passes `policy-*` headers to the `_administrators` bucket in the Java and C# SDKs.
- The C# SDK now supports chunked reads for a POST response and returns the trailer headers from the response in the `ScspResponse` object headers. The C# SDK supports multi-part completion POST.
- Support for Content Router enumeration is removed from the C# SDK as Content Router is deprecated.
- The Python SDK client now supports chunked reads on POST responses, including for multi-part completion POSTs.

SDK version 6.1.4

This release includes the following enhancements and changes:

- The Java SDK now builds using Maven and depends on HttpClient 4.2.5.
- Support for Content Router enumeration is removed from the Java SDK as Content Router is deprecated.

SDK version 6.1.3

This release includes the following enhancements and changes:

- The C# SDK now correctly handles empty trailer headers on a chunked encoding response.
- The C# SDK now includes a ConnectionPool instance to allow connection sharing between requests. **Important:** When finished with an SCSPClient instance, applications must now explicitly call Close to verify connections are not kept open.

SDK version 6.1.2

This release includes the following enhancements and changes:

- This release contains performance refactoring and optimizations for Expect/Continue handling in .Net for the C# client. The pattern closely matches Swarm reference implementations and has shown significant improvements in throughput, correctness and transaction rates in testing.

Limitations

These are the known issues and operational limitations that exist in this release of the Swarm SDK.

- **Supported operating systems.** English versions of operating systems are supported. Other versions or distributions, including languages other than English, are not currently supported.
- **Swarm Locator** Some languages like Java may include examples for how to use other locators like mDNS but these are considered examples and need to be independently tested and verified.
- **C++ Integrity Seal Hash Upgrades** Due to an issue with the way cURL handles long trailer headers, upgrading an integrity seal hash with the C++ client can fail occasionally.
- **Using Range headers.** Java, C++, and Python language implementations enable specifying a Range header without bytes=, which is in violation of [RFC 2616, section 3.12](#). The C# implementation does not have this issue. Code examples provided with each language show the correct way to specify a Range header. See the headers.AddRange example in the RunReadExamples method.

Deprecation Notices

This section lists functions that are deprecated and are subject to being removed in future SDK releases.

- The functions NoCastorNodesLocatorError.getFriendlyError and getFriendlyError in proxyLocator.py are removed.
- Deprecated in SDK version 1.2: The uuid parameter is replaced by path. Examples from SDK sample code follow:
- Java:
 - SDK 1.1: ScspResponse rcResponse = client.readMutable(uuid, "", outputStream, args, new ScspHeaders());
 - SDK 1.2 and later: ScspResponse rcResponse = client.readMutable("", uuid, outputStream, args, new ScspHeaders());
- Python:

- SDK 1.1: `rcResponse = client.readMutable(uuid, fread, None, None)`
- SDK 1.2 and later: `rcResponse = client.readMutable("", fread, None, None, path=uuid)`
- C++:
 - SDK 1.1: `client.readMutable(uuid, &outputStream, &response);`
 - SDK 1.2 and later: `client.readMutable("", &outputStream, &response, NULL, NULL, uuid);`
- C#:
 - SDK 1.1: `ScspResponse rcResponse = client.ReadMutable(uuid, "", outputStream, args, new ScspHeaders());`
 - SDK 1.2 and later: `ScspResponse rcResponse = client.ReadMutable("", uuid, outputStream, args, new ScspHeaders());`

Application and Configuration Notes

Pay special attention to the following items when developing Swarm client implementations.

- **ScspClient chunkSize parameters support in C++.** ScspClient in all languages, including C++, supports the following parameters: `getChunkSize`, `setChunkSize`. cURL does not support explicitly setting how many bytes are sent at a time. cURL provides a buffer and the buffer's length but does not enable the SDK to set the size of the buffer.
- **C# Write, Update, Append.** A Write, Update, or Append using the C# SDK client that encounters an error response, an `ScspWebException` may be thrown. This can occur with a 400 response from the cluster, or on any error response (code 400 and greater) when using the SCSP Proxy. This behavior is caused by the way that .Net internally handles a connection closing while writing data to a peer.
- **Java recompile required.** Because of internal changes to the Java SDK client, the Java code must be recompiled against the classes provided with the SDK.
- **C# connection timeout.** Increasing the connection timeout may help alleviate write failures on large objects due to too many retries of cancelled requests. For more information, see the chapter on C# in the SDK Overview.
- **Preemptive authorization** (from [RFC2617](#)). Preemptive authorization enables client applications to generate an authorization header initially, bypassing the server's authentication challenge. Language-specific implementation details for the Swarm SDK follow:
 - **Python and C++.** Preemptive authentication works.
 - **Java.** Preemptive authorization does not work. Every request for a protected resource generates an initial 401 (Unauthorized) response from Swarm.
 - **C#.** Preemptive authentication fails with requests requiring authentication in different domains.
- Build cURL using Visual Studio C++ 2008 or earlier for best results. Problems building the latest cURL version may occur with Visual Studio C++ 2010. See the [cURL install page](#) for more information.
- **Failed Integrity Seal Validation** Swarm closes the connection if an integrity seal fails validation on a read, which is shown in the client as an I/O error.
- **C++ character encoding.** The string class must be used when passing in a URI path as a string using the C++ SDK. These characters must be UTF-8 encoded by the caller if the path needs includes non-ASCII characters.
- **Java, C++, C#, and Python character encoding.** Escape a backslash character (`\`) with `%5c` when passing in a URI path using the SDK.
- **Java ResettableFileInputStream** In the Java client, `FileInputStream()` cannot be used; instead, use `ResettableFileInputStream`, which is located in `CAStorSDK-src\com \caringo\client`.

SwarmFS Release Notes

SwarmNFS renaming

SwarmNFS is renamed SwarmFS as of Swarm 12 to reflect the expanded scope of capabilities.

SwarmFS is a lightweight file protocol converter that brings the benefits of Swarm's scale-out object storage to NFSv4, seamlessly integrating files and object storage. With SwarmFS, data can be securely stored and accessed via NFSv4, S3, HDFS, and SCSP/HTTP, making it possible to organize billions of files coming from different protocols, distribute data to different locations, and search all files at once.

SwarmFS brings the benefits of native object storage to NFS, but it is not a complete replacement for all traditional file (NAS/SAN) needs. In particular, note the following:

- **Rapid updates** – Frequent file updates, such as updates to databases, video editing, and storage of active vmdk files does not perform well and are not recommended.
- **Large files** – SwarmFS performs well with files up to 30GB and best with files of 10GB and smaller; writing files of 100GB and greater is regularly tested. Reading from files (objects) of any size is fully supported.
- **Versioning** – Object versioning is supported (the last version written by any method becomes the current version), but it generates heavy demands on storage resources. Enabling it in a SwarmFS context is not recommended.
- [SwarmFS 3 Releases](#)
- [SwarmFS 1.2 Release](#)
- [SwarmFS 2 Releases](#)

SwarmFS 3 Releases

- [SwarmFS 3.2](#)
- [SwarmFS 3.1](#)
- [SwarmFS 3.0](#)

SwarmFS 3.2

Third party software packages are updated.

The prior **Upgrading** and **Known Issues** sections below apply.

SwarmFS 3.1

With the 3.1 release, SwarmFS gains improvements in metrics initialization and timestamps for copies. (NFS-837, NFS-836, NFS-835)

With the Swarm 12.0 release, SwarmFS object uploads that are stalled “in progress” now timeout to allow consolidation and clean up of the uploaded parts. (SWAR-7699)

The prior **Upgrading** and **Known Issues** sections below apply.

SwarmFS 3.0

With the 3.0 release, SwarmFS removes dependency on Elasticsearch versioning and uses the folder listing service in Content Gateway 7.0.

With version 7.0 of Gateway, *folder listing* support is completely rearchitected and centralized within Content Gateway, retiring the legacy folder listing in SwarmFS. Folder listing allows SwarmFS to render virtual folders *below* the bucket level of Swarm Storage: it translates any delimited prefixes in Swarm object names (such as in `FY2019/Q3/object.jpg`) into folders on your users' file systems. See [SwarmFS Listings](#).

The new architecture brings many benefits to SwarmFS:

- Future upgrades of SwarmFS are free of dependency on the version of Elasticsearch (once on version 6 or higher).
- Elasticsearch security is strengthened, with listing queries being locked down to the domain/bucket.
- Elasticsearch clusters are now free to move to more protected network locations, now that listing no longer requires direct access to Elasticsearch.
- Authorization is now centralized through Gateway and verifies users see data within the tenant/domain/bucket. Prior versions cannot use [Content Gateway Authentication](#) because they accessed Elasticsearch directly.
- The pagination of large listing results is no longer bound to the Elasticsearch limit (`index.max_result_window`).
- The listing service uses features new to Elasticsearch 6.

The scope of this release does not include unnamed objects, caching, folder locking/leasing, or client notification of namespace changes.

Upgrading

Best practice is to upgrade to Elasticsearch 6 and Gateway 7.0, which is the platform that supports the new listing service and removes dependency on versions of Elasticsearch. A critical error is logged if SwarmFS runs with a version of Gateway older than 6.4.

1. Follow the guidance in [SwarmFS Deployment](#) for what specific configuration is required across components.
2. Complete the section for SwarmFS when migrating Elasticsearch: [Migrating from Older Elasticsearch](#).

Known Issues

- If, instead of updating, you perform a yum remove of SwarmFS and also remove its artifacts ("`rm -rf /etc/ganesha`"), the configuration (`/etc/ganesha/ganesha.conf`) is not recreated on install, causing the SwarmFS-config script to fail. Workaround: Save the `ganesha.conf` and restore it to that directory. (NFS-778)
- If application file handling fails to clean up after unlinked files, 'silly' files (of form `.nfsXXXX`) may persist in directories, consuming space. Workaround: Add a cron job that periodically looks for and removes such files. (NFS-764)
- Do not use SwarmFS with a bucket that has versioning enabled. File writes can commit the object multiple times, resulting in an excessive number of versions. (NFS-753)
- Externally-written custom headers may not appear in `:metadata` reads. Workaround: To trigger ES to pick up an external update, also set the `X-Data-Modified-Time-Meta` header to the current time (in seconds since epoch). (NFS-692)
- Exports defined with different domains but the same bucket name do not operate as unique exports. (NFS-649)
- An invalid bucket name entered for an export in the UI silently fails in SwarmFS (config reads, export generates, client mounts, 0-byte writes and directory operations appear to succeed) but fails on requests to Swarm Storage. (NFS-613)
- The SwarmFS configuration script does not work with config URLs that use HTTPS and contain auth credentials for accessing Swarm through Gateway. (NFS-406)
- On startup, SwarmFS may generate erroneous and harmless WARN level messages for configuration file parameters, such as `config_errs_to_log :CONFIG :WARN :Config File (/etc/ganesha/ganesha.conf:17): Unknown parameter (Path)` (NFS-289)

SwarmFS 1.2 Release

SwarmFS 1.2 must be used with a Swarm cluster running Storage 9.3.1+ and with Storage UI 1.2.1.

New Features and Changes

- Symbolic links (soft) are now supported.
- Demo clusters or those running on slower hardware or VMs are now supported.
 - Because slower hardware/VMs may require a longer update delay to operate correctly, the configuration now includes the setting `Scsp/UpdateDelay`.
 - See the *Implementation Notes* in [SwarmFS Server Installation](#).
- Fixed: Symbolic links to files did not return metadata if read using the ".metadata" suffix. (SNFS-346)
- Fixed: Generating core files can now be enabled and disabled via the `nfs-ganesha.service` file or through the system-wide configuration. (SNFS-297)

Known Issues

- Directory listings may appear to hang but complete successfully when large writes are in progress.
- Accessing unnamed objects is not supported.

SwarmFS 2 Releases

- [SwarmFS 2.4](#)
- [SwarmFS 2.3](#)
- [SwarmFS 2.2](#)
- [SwarmFS 2.1](#)
- [SwarmFS 2.0.2](#)
- [SwarmFS 2.0.1](#)
- [SwarmFS 2.0.0](#)

SwarmFS 2.4

With the 2.4 release, SwarmFS adds support for Swarm 11 and Elasticsearch 6.

- Added support for Elasticsearch 6.8.6. (NFS-808)
- Swarm NFS 2.4 supports Swarm Storage 11.0 and higher. (NFS-804)

Use the supported versions of Swarm components for the target version of Elasticsearch:

SwarmFS 2.4	Elasticsearch 6.8.6	Swarm Storage 11.1	Gateway 6.3
	Elasticsearch 5.6.12	Swarm Storage 10.0 - 11.1	Gateway 6.0 - 6.3
SwarmFS 2.1	Elasticsearch 2.3.3	Swarm Storage 10.0 - 11.1	Gateway 5.4

Upgrading

1. Follow the guidance in [SwarmFS Deployment](#) for what specific configuration is required across components.
2. Complete the section for SwarmFS when migrating Elasticsearch: [Migrating from Older Elasticsearch](#).

Known Issues

- If, instead of updating, perform a yum remove of SwarmFS and also remove the artifacts ("rm -rf /etc/ganesha"), the configuration (/etc/ganesha/ganesha.conf) is not recreated on install, causing the SwarmFS-config script to fail. Workaround: Save the ganesh.conf and restore it to that directory. (NFS-778)
- 'silly' files (of form `.nfsXXXX`) may persist in directories, consuming space if application file handling fails to clean up after unlinked files. Workaround: Add a cron job that periodically looks for and removes such files. (NFS-764)
- Do not use SwarmFS with a bucket that has versioning enabled. File writes can commit the object multiple times, resulting in an excessive number of versions. (NFS-753)
- Externally-written custom headers may not appear in :metadata reads. Workaround: To trigger ES to pick up an external update, also set the `X-Data-Modified-Time-Meta` header to the current time (in seconds since epoch). (NFS-692)
- Exports defined with different domains but the same bucket name do not operate as unique exports. (NFS-649)
- An invalid bucket name entered for an export in the UI fails silently in SwarmFS (config reads, export generates, client mounts, 0-byte writes and directory operations appear to succeed) but fails on requests to Swarm Storage. (NFS-613)
- The SwarmFS configuration script does not work with config URLs using HTTPS and contain auth credentials for accessing Swarm through Gateway. (NFS-406)
- On startup, SwarmFS may generate erroneous and harmless WARN level messages for configuration file parameters, such as `config_errs_to_log :CONFIG :WARN :Config File (/etc/ganesha/ganesha.conf:17): Unknown parameter (Path)` (NFS-289)
- SwarmFS supports exclusive opens of a file (`O_EXCL` and `O_CREATE`) but does not support exclusive reopens (`EXCLUSIVE4`). (NFS-69)
- To prevent problems resulting from SwarmFS disconnects or shutdowns, the Storage setting `health.parallelWriteTimeout` must be set to a non-zero value, such as 1209600 (2 weeks). (NFS-63)

SwarmFS 2.3

With the 2.3 release, SwarmFS includes several fixes. This release requires Gateway 6.0 with Elasticsearch 5.6, on Swarm 10. Remain on version 2.1 if still using Gateway 5.4 with Elasticsearch 2.3.3.

- Credentials in the JSON file for exports are now handled via HTTPS, so they are encrypted during transmission. Note: credentials within the `ganesha.conf` file must be protected at the file-system level. (NFS-790)
- SwarmFS has improved support for Windows clients by allowing empty directories to be created and immediately renamed, as happens with Windows File Explorer. (NFS-789)
- SwarmFS now has a mechanism to prevent shares from mounting before content can be served. To enable this feature, add the new parameter, `ExportAfterGrace = TRUE;`, to the `ganesha.conf` file. (NFS-787)
- Fixed: RHEL/CentOS 7.6 clients exhibited problems mounting SwarmFS 2.2. (NFS-781)
- Fixed: For export configurations, the `defaultrootowner` / `defaultrootgroup` and permission mode in octal are not being set correctly in the UI, and the link count is incorrect in the export directory inode. (NFS-783)

Known Issues

- Do not use Swarm NFS 2.3 with Swarm 11.0. (NFS-804)
- If, instead of updating, perform a `yum remove` of SwarmFS and also remove the artifacts ("`rm -rf /etc/ganesha`"), the configuration (`/etc/ganesha/ganesha.conf`) is not recreated on install, causing the SwarmFS-config script to fail. Workaround: Save the `ganesha.conf` and restore it to that directory. (NFS-778)
- 'silly' files (of form `.nfsXXXX`) may persist in directories, consuming space if application file handling fails to clean up after unlinked files. Workaround: Add a cron job that periodically looks for and removes such files. (NFS-764)
- Do not use SwarmFS with a bucket that has versioning enabled. File writes can commit the object multiple times, resulting in an excessive number of versions. (NFS-753)
- Externally-written custom headers may not appear in `:metadata` reads. Workaround: To trigger ES to pick up an external update, also set the `X-Data-Modified-Time-Meta` header to the current time (in seconds since epoch). (NFS-692)
- Exports defined with different domains but the same bucket name do not operate as unique exports. (NFS-649)
- An invalid bucket name entered for an export in the UI fails silently in SwarmFS (config reads, export generates, client mounts, 0-byte writes and directory operations appear to succeed) but fails on requests to Swarm Storage. (NFS-613)
- The SwarmFS configuration script does not work with config URLs using HTTPS and contain auth credentials for accessing Swarm through Gateway. (NFS-406)
- On startup, SwarmFS may generate erroneous and harmless WARN level messages for configuration file parameters, such as `config_errs_to_log :CONFIG :WARN :Config File (/etc/ganesha/ganesha.conf:17): Unknown parameter (Path)` (NFS-289)
- SwarmFS supports exclusive opens of a file (`O_EXCL` and `O_CREATE`) but does not support exclusive reopens (`EXCLUSIVE4`). (NFS-69)
- To prevent problems resulting from SwarmFS disconnects or shutdowns, the Storage setting `health.parallelWriteTimeout` must be set to a non-zero value, such as 1209600 (2 weeks). (NFS-63)

SwarmFS 2.2

With the 2.2 release, SwarmFS now fully supports and requires Gateway 6.0 with Elasticsearch 5.6, on Swarm 10.



Required

Remain on version 2.1 while using Gateway 5.4 with Elasticsearch 2.3.3.

Known Issues

- RHEL/CentOS 7.6 clients exhibit problems mounting SwarmFS. Do not upgrade to this version until this issue is resolved. (NFS-781)
- If, instead of updating, perform a yum remove of SwarmFS and also remove the artifacts ("rm -rf /etc/ganesha"), the configuration (/etc/ganesha/ganesha.conf) is not recreated on install, causing the SwarmFS-config script to fail. Workaround: Save the ganesha.conf and restore it to that directory. (NFS-778)
- 'silly' files (of form `.nfsXXXX`) may persist in directories, consuming space if application file handling fails to clean up after unlinked files. Workaround: Add a cron job that periodically looks for and removes such files. (NFS-764)
- Do not use SwarmFS with a bucket that has versioning enabled. File writes can commit the object multiple times, resulting in an excessive number of versions. (NFS-753)
- Externally-written custom headers may not appear in `:metadata` reads. Workaround: To trigger ES to pick up an external update, also set the `X-Data-Modified-Time-Meta` header to the current time (in seconds since epoch). (NFS-692)
- Exports defined with different domains but the same bucket name do not operate as unique exports. (NFS-649)
- An invalid bucket name entered for an export in the UI fails silently in SwarmFS (config reads, export generates, client mounts, 0-byte writes and directory operations appear to succeed) but fails on requests to Swarm Storage. (NFS-613)
- On startup, SwarmFS may generate erroneous and harmless WARN level messages for configuration file parameters, such as `config_errs_to_log :CONFIG :WARN :Config File (/etc/ganesha/ganesha.conf:17): Unknown parameter (Path)` (NFS-289)
- SwarmFS supports exclusive opens of a file (`O_EXCL` and `O_CREATE`) but does not support exclusive reopens (`EXCLUSIVE4`). (NFS-69)
- To prevent problems resulting from SwarmFS disconnects or shutdowns, the Storage setting `health.parallelWriteTimeout` must be set to a non-zero value, such as 1209600 (2 weeks). (NFS-63)

SwarmFS 2.1

New Features and Changes

- To generate performance data, SwarmFS now has profile logging, which is a configuration option disabled by default and hidden from the UI. Enable this logging as directed by DataCore Support: once logs are generated, send them to DataCore Support, which have tools to analyze the read performance. (NFS-719)
- SwarmFS has significantly improved the performance of sequential reads. (NFS-714)
- Logging for audit purposes is improved. Open, delete, and rename operations generate NIV_EVENT-level messages in the standard SwarmFS log. (NFS-684)
- Define default Owner, Group, and ACL to apply to any objects and synthetic folders created externally without preset POSIX permissions attached via metadata when configuring SwarmFS exports. (NFS-610)
- SwarmFS now has a global hard/soft memory limit to work in conjunction with each export's own configured limits, to make better use of NFS server resources. Multiple exports on a single server now share the globally allotted buffer memory, rather than each carving out a separate private buffer memory. (NFS-511)
- SwarmFS supports the Linux `cp` command for copying metadata (`cp file1:metadata file2:metadata`) and data (`cp file1:data file2:data`), creating a new destination file with 0 bytes if needed. (NFS-469)

Known Issues

- Externally-written custom headers may not appear in `:metadata` reads. Workaround: To trigger ES to pick up an external update, also set the `X-Data-Modified-Time-Meta` header to the current time (in seconds since epoch). (NFS-692)
- Exports defined with different domains but the same bucket name do not operate as unique exports. (NFS-649)
- An invalid bucket name entered for an export in the UI fails silently in SwarmFS (config reads, export generates, client mounts, 0-byte writes and directory operations appear to succeed) but fails on requests to Swarm Storage. (NFS-613)
- On startup, SwarmFS may generate erroneous and harmless WARN level messages for configuration file parameters, such as `config_errs_to_log :CONFIG :WARN :Config File (/etc/ganesha/ganesha.conf:17): Unknown parameter (Path)` (NFS-289)
- SwarmFS supports exclusive opens of a file (`O_EXCL` and `O_CREATE`) but does not support exclusive reopens (`EXCLUSIVE4`). (NFS-69)
- To prevent problems resulting from SwarmFS disconnects or shutdowns, the Storage setting `health.parallelWriteTimeout` must be set to a non-zero value, such as 1209600 (2 weeks). (NFS-63)

SwarmFS 2.0.2

- Fixed: Issues existed with directories including spaces in names. (NFS-593)

SwarmFS 2.0.1

SwarmFS 2.0.1 must be used with a Swarm cluster running Storage 9.5+ and with Storage UI 1.2.4.

New Features and Changes

- Performance is improved for how quickly external object updates appear in SwarmFS listings.

Known Issues

- An invalid bucket name entered for an export in the UI fails silently in SwarmFS (config reads, export generates, client mounts, 0-byte writes and directory operations appear to succeed) but fails on requests to Swarm Storage. (NFS-613)
- Cloud Security Authentication type **Session Token** is not available, although it appears as an option in the export definition.
- Reading metadata over NFS using `{filename}:metadata` is supported, but editing of object metadata over NFS is not supported.
- To prevent problems resulting from SwarmFS disconnections or shutdowns, the Storage setting `health.parallelWriteTimeout` must be set to a non-zero value, such as 1,209,600 (2 weeks). (NFS-63)
 - Note: changing this setting affects S3, which defaults to keeping uncompleted multipart uploads indefinitely.
- To use SwarmFS with Storage 9.5.0, set `scsp.keepAliveInterval = 45`. For best results, set **Request timeout** for each export to 90, so it is at least twice the value of `scsp.keepAliveInterval`. (NFS-535, SWAR-7917)

SwarmFS 2.0.0

SwarmFS 2.0.0 must be used with a Swarm cluster running Storage 9.5+ and with Storage UI 1.2.3.

New Features and Changes

- Swarm Content Gateway is now supported. The SwarmFS export configuration in Storage UI now supports Content Gateway in addition to Direct to Swarm. The **Cloud Security** section of each export configuration allows setting up the method that best fits the situation: Session Token (token admin credentials with expiration), Single User (user, password, and token), or Pass-through. See [SwarmFS Export Configuration](#).
- The defaults for NFS timeouts are shortened to improve error handling. See [SwarmFS Export Configuration](#). (UIS-775)

Known Issues

- The default timeouts must be increased when creating an export in the UI: in the **Advanced Settings**, set the Retries Timeout, Request Timeout, and Write Timeout all to 90 seconds.
- Cloud Security Authentication type **Session Token** is not available, although it appears as an option in the export definition.
- Reading metadata over NFS using `{filename}:metadata` is supported, but editing of object metadata over NFS is not supported.
- To prevent problems resulting from SwarmFS disconnections or shutdowns, the Storage setting `health.parallelWriteTimeout` must be set to a non-zero value, such as 1,209,600 (2 weeks). (NFS-63)
- To use SwarmFS with Storage 9.5.0, set `scsp.keepAliveInterval = 45`. For best results, set **Request timeout** for each export to 90, so it is at least twice the value of `scsp.keepAliveInterval`. (NFS-535, SWAR-7917)
- Issues exist with feeds defined to use a non-default admin password. (UIS-759)
- Accessing unnamed objects is not supported.

Swarm Storage Release Notes

Important

If you are upgrading from a prior version, review the changes and upgrade impacts for *each version* since the version from which you are upgrading.

- [Swarm Storage 14.1 Release](#)
- [Swarm Storage 14.0.1 Release](#)
- [Swarm Storage 12.1 Release](#)
- [Swarm Storage 12.0 Release](#)
- [Swarm Storage 11 Releases](#)

Swarm Storage 14.1 Release

- [New Features](#)
- [Additional Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Known Issues](#)

New Features

- The DELETE operation returns the `Castor-System-Delete-Marker-Etag` header on a named or alias object in the versioning enabled or versioning suspended state. If versioning is disabled, the object is immutable or a context, and no such header is returned. The ETag of the delete marker (returned) is checked by INFO-ing the deleted object and checking the returned ETag header. (SWAR-9292)
- Updated the DELETE behavior of the current object version that contains a non-deletion lifepoint. The behavior depends on the container's versioning state of an object and refers to the bucket for a named object. For alias tenanted objects, it is a domain's "unnamed" versioning policy. (SWAR-9265)
 - If the container is versioning disabled or versioning suspended, deletion of the current version fails, as before, with a 403.
 - If the container is versioning enabled, the delete succeeds and creates a delete marker, leaving the former current version as the first non-current version.
- Swarm provides a setting (`ec.convertToPolicy`) to enable the conversion of existing EC objects to the current policy encoding. (SWAR-9203)

Additional Changes

Changes include versions and fixes coming from testing and user feedback:

- **OSS Versions** – See [Third-Party Components for Storage 14.1](#) for the complete listing of packages and versions for this release.
- **Fixed in 14.1**
 - **Replication and S3 backup feed:** The feed is no longer reported as blocked for long-running replications (SWAR-9218). The object replication is allowed for custom metadata containing Non-ASCII characters. (SWAR-9212)
 - **Bucket and domain renaming:** Swarm supports renaming of query arguments and bucket and domain using alias UUID. Renaming works with COPY, PUT, and APPEND operations, though COPY with the preserve query argument is advised. If a bucket or domain collision is detected, rename either context without colliding with each other. (SWAR-8708)
 - **Bulk configuration:** Swarm provides bulk Elasticsearch 7 nodes configuration to consistently use `discovery.seed_hosts`, not older `discovery.zen` settings. (SWARM-9395)
 - **nicTable:** It is populated again in SNMP; multiple new columns were added. (SWAR-9327)
 - **SNMP certain communication stats:** Swarm now publishes via SNMP communication stats summarized over a moving window covering the last hour. A new daily stats table in SNMP publishes both communication and network interface stats summarized over a moving window covering the last day. The daily stats are updated every hour. (SWAR-8124)
 - **Large physical memory support:** The EFQ02 errors are not received in logs while running 14.1 release on nodes with large physical memory (>116G). (SWAR-9353)
 - **NTP dependency:** NTP dependency checks are improved for configured time servers that either fail to respond quickly or respond using an older NTP version. (SWAR-9373)
 - **USB Booting** - Swarm 14.1.1 now supports booting from a USB drive. This was not working in 14.1.0. (SWAR-9419)

Upgrade Impacts

Required

Complete the migration to Swarm 11.3 and ES 6.8.6 before upgrading to Swarm 14 if running older Elasticsearch (5.6.12 or 2.3.3). See [here](#) for upgrading from an unsupported Elasticsearch version.

These items are changes to the product function that may require operational or development changes for integrated applications. Address the upgrade impacts for each of the versions since the currently running version:

Impacts for 14.1

- **Swarm storage node metrics are deprecated** and are replaced in the next major release by the graphs and reporting from [Grafana and Prometheus Node Exporter](#). The storage administration UI is updated to allow for metrics to be turned off. Clear `metrics.target` from the configuration, uninstall `caringo-elasticsearch-metrics`, and `curl -XDELETE 'http://ELASTICSEARCH:9200/metrics-*` to clear the space in the Elasticsearch cluster. (SWAR-8982)
- **Differences in `scsp.forceLegacyNonce` configuration** depends on the version upgrading from. (SWAR-9020)
 - **Currently running a Swarm Storage version prior to 11.1**, and upgrading to 11.1, 11.2, 11.3, 12.0 or 12.1:

Before upgrading, set `scsp.forceLegacyNonce=true` in the `node.cfg` file. After the upgrade, when the cluster is fully up, update `scsp.forceLegacyNonce=false` using `swarmctl` and change `scsp.forceLegacyNonce=false` in the `node.cfg` file.
 - **Currently running a Swarm Storage version 11.1, 11.2, 11.3, 12.0 or 12.1** and upgrading to another version from that list:

Before upgrading, verify `scsp.forceLegacyNonce=false` is in the `node.cfg` file using `swarmctl` that `scsp.forceLegacyNonce=false` is in a cluster.
 - **Currently running a Swarm storage version 11.1, 11.2, 11.3, 12.1 or 14.0** and upgrading to 14.1 or later:

Remove `scsp.forceLegacyNonce` from the `node.cfg` file.

Use `swarmctl` to check or change settings

Use `'swarmctl -C scsp.forceLegacyNonce'` to check the value of `scsp.forceLegacyNonce`.

Use `'swarmctl -C scsp.forceLegacyNonce -V False'` to set the value to `false`.

For more details, see <https://support.cloud.caringo.com/tools/Tech-Support-Scripts-Bundle-swarmctl.pdf>.

Cumulative impacts

Address all upgrade impacts for *each version released* since the version being upgraded from.

Watch Items and Known Issues

The following watch items are known:

- A node fails to mount all disks in the node if the node mounts an encrypted volume with a missing encryption key in the configuration. (SWAR-8762)
- The chassis shuts down but does not come back up when restarting a cluster of virtual machines that are UEFI-booted (versus legacy BIOS). (SWAR-8054)
- Lifecycle policy `NamePrefix` rules (for Non-ASCII characters) do not match the intended objects, hence, the rule does not fire. (SWAR-9413)
- Dates used in lifecycle policy rules do not include a time specifier or time zone specifier. Use YYYY-MM-DD or YYYYMMDD formatted dates only. This restriction will be relaxed in later releases. (SWAR-9408)

- Verify the configured "*java.io.tmpdir*" in "*jvm.options*" is writable to Elasticsearch for customers using Elasticsearch instances that fail to start with JNA warnings in Elasticsearch logs. Change "*java.io.tmpdir*" to `/var/log/elasticsearch` as per desired security preferences. (SWAR-9347)
- Swarm versions 10.0 onward are vulnerable to a kernel issue manifested on some Intel CPUs. Symptoms include lowered performance, long mount times, and cluster instability. Swarm 14.1 provides a work-around for this issue, see [Intel Skylake/Cascade Lake CPU Performance Issue](#). (SWAR-9055)

These are standing operational limitations:

- The Storage UI shows no NFS config if the Elasticsearch cluster is wiped. Contact DataCore Support for help in repopulating the SwarmFS config information. (SWAR-8007)
- Any incomplete multipart upload into a bucket leaves the parts (unnamed streams) in the domain if a bucket is deleted. To find and delete those parts, use the `s3cmd` utility (search the Support site for "`s3cmd`" guidance). (SWAR-7690)
- Invalid config parameters that prevent the unassigned nodes from booting are created if subcluster assignments are removed in the CSN UI. (SWAR-7675)

To upgrade Swarm 9 or higher, proceed to [How to Upgrade Swarm](#). For migration from Swarm 8.x or earlier, contact DataCore Support for guidance.

Third-Party Components for Storage 14.1

For licensing information, see [Open Source Software Licenses](#).

Elasticsearch and Swarm distributions

Elasticsearch 7.5.2
 Swarm S3 Backup Restore 1.2.4
 Swarm Search 7.0.1
 Swarm Metrics 7.0.1

Swarm Storage components

Operating system: Debian GNU/Linux 11 (bullseye)
 Linux kernel: 5.14.9
 kernel module 3w_9xxx 3ware 9000 Storage Controller Linux Driver: 2.26.02.014
 kernel module 3w_sas LSI 3ware SAS/SATA-RAID Linux Driver: 3.26.02.000
 kernel module 3w_xxxx 3ware Storage Controller Linux Driver: 1.26.02.003
 kernel module 8021q : 1.8
 kernel module 8139cp RealTek RTL-8139C+ series 10/100 PCI Ethernet driver: 1.3
 kernel module 8139too RealTek RTL-8139 Fast Ethernet driver: 0.9.28
 kernel module aacraid Dell PERC2, 2/Si, 3/Si, 3/Di, Adaptec Advanced Raid Products, HP NetRAID-4M, IBM ServeRAID & ICP SCSI driver: 1.2.1
 [50983]-custom
 kernel module acard_ahci ACard AHCI SATA low-level driver: 1.0
 kernel module ad7418 AD7416/17/18 driver: 0.4
 kernel module ahci AHCI SATA low-level driver: 3.0
 kernel module aic79xx Adaptec AIC790X U320 SCSI Host Bus Adapter driver: 3.0
 kernel module aic7xxx Adaptec AIC77XX/78XX SCSI Host Bus Adapter driver: 7.0
 kernel module aic94xx Adaptec aic94xx SAS/SATA driver: 1.0.3
 kernel module am53c974 AM53C974 SCSI driver: 1.00
 kernel module arcmsr Areca ARC11xx/12xx/16xx/188x SAS/SATA RAID Controller Driver: v1.50.00.05-20210429
 kernel module ata_generic low-level driver for generic ATA: 0.2.15
 kernel module ata_piix SCSI low-level driver for Intel PIIX/ICH ATA controllers: 2.13
 kernel module atxp1 System voltages control via Attansic ATXP1: 0.6.3
 kernel module be2iscsi Emulex OneConnectOpen-iSCSI Driver version11.4.0.1 Driver 11.4.0.1: 11.4.0.1
 kernel module bfa QLogic BR-series Fibre Channel HBA Driver fcpim: 3.2.25.1
 kernel module bnx2fc QLogic FCoE Driver: 2.12.13
 kernel module bnx2i QLogic NetXtreme II BCM5706/5708/5709/57710/57711/57712/57800/57810/57840 iSCSI Driver: 2.7.10.1
 kernel module cnic QLogic cnic Driver: 2.5.22
 kernel module csiostor Chelsio FCoE driver: 1.0.0-ko
 kernel module cxgb3i Chelsio T3 iSCSI Driver: 2.0.1-ko
 kernel module cxgb4i Chelsio T4-T6 iSCSI Driver: 0.9.5-ko
 kernel module dca : 1.12.1
 kernel module eeprom_93cx6 EEPROM 93cx6 chip driver: 1.0
 kernel module efivars sysfs interface to EFI Variables: 0.08
 kernel module esas2r esas2r: 1.00
 kernel module esp_scsi ESP SCSI driver core: 2.000
 kernel module fnic Cisco FCoE HBA Driver: 1.6.0.53
 kernel module hpsa Driver for HP Smart Array Controller version 3.4.20-200: 3.4.20-200
 kernel module i40e Intel(R) 40-10 Gigabit Ethernet Connection Network Driver: 2.17.4
 kernel module ioatdma : 5.00

kernel module ipmi_msghandler Incoming and outgoing message routing for an IPMI interface.: 39.2
kernel module ipr IBM Power RAID SCSI Adapter Driver: 2.6.4
kernel module ips IBM ServerRAID Adapter Driver 7.12.05: 7.12.05
kernel module iscsi : 1.2.0
kernel module jme JMicron JMC2x0 PCI Express Ethernet driver: 1.0.8
kernel module libcxgbi Chelsio iSCSI driver library: 0.9.1-ko
kernel module lpc Emulex LightPulse Fibre Channel SCSI driver 12.8.0.10: 0
kernel module megaraid LSI Logic MegaRAID legacy driver: 2.00.4
kernel module megaraid_mbox LSI Logic MegaRAID Mailbox Driver: 2.20.5.1
kernel module megaraid_mm LSI Logic Management Module: 2.20.2.7
kernel module megaraid_sas Broadcom MegaRAID SAS Driver: 07.717.02.00-rc1
kernel module mlx4_core Mellanox ConnectX HCA low-level driver: 4.0-0
kernel module mlx4_en Mellanox ConnectX HCA Ethernet driver: 4.0-0
kernel module mpt3sas LSI MPT Fusion SAS 3.0 Device Driver: 37.101.00.00
kernel module mptbase Fusion MPT base driver: 3.04.20
kernel module mptctl Fusion MPT misc device (ioctl) driver: 3.04.20
kernel module mptfc Fusion MPT FC Host driver: 3.04.20
kernel module mptsas Fusion MPT SAS Host driver: 3.04.20
kernel module mptscsih Fusion MPT SCSI Host driver: 3.04.20
kernel module mptspi Fusion MPT SPI Host driver: 3.04.20
kernel module mtip32xx Micron RealSSD PCIe Block Driver: 1.3.1
kernel module mvsas Marvell 88SE6440 SAS/SATA controller driver: 0.8.16
kernel module myri10ge Myricom 10G driver (10GbE): 1.5.3-1.534
kernel module ne2k_pci PCI NE2000 clone driver: 1.03
kernel module netxen_nic QLogic/NetXen (1/10) GbE Intelligent Ethernet Driver: 4.0.82
kernel module nicpf Cavium Thunder NIC Physical Function Driver: 1.0
kernel module nicvf Cavium Thunder NIC Virtual Function Driver: 1.0
kernel module niu NIU ethernet driver: 1.1
kernel module nvme : 1.0
kernel module nvme_core : 1.0
kernel module pata_acpi SCSI low-level driver for ATA in ACPI mode: 0.2.3
kernel module pata_ali low-level driver for ALi PATA: 0.7.8
kernel module pata_amd low-level driver for AMD and Nvidia PATA IDE: 0.4.1
kernel module pata_artop SCSI low-level driver for ARTOP PATA: 0.4.6
kernel module pata_atiixp low-level driver for ATI IXP200/300/400: 0.4.6
kernel module pata_atp867x low level driver for Artop/Acard 867x ATA controller: 0.7.5
kernel module pata_cmd64x low-level driver for CMD64x series PATA controllers: 0.2.18
kernel module pata_efar SCSI low-level driver for EFAR PIIX clones: 0.4.5
kernel module pata_hpt366 low-level driver for the Highpoint HPT366/368: 0.6.11
kernel module pata_hpt37x low-level driver for the Highpoint HPT37x/30x: 0.6.23
kernel module pata_hpt3x2n low-level driver for the Highpoint HPT3xxN: 0.3.15
kernel module pata_hpt3x3 low-level driver for the Highpoint HPT343/363: 0.6.1
kernel module pata_it821x low-level driver for the IT8211/IT8212 IDE RAID controller: 0.4.2
kernel module pata_jmicron SCSI low-level driver for Jmicron PATA ports: 0.1.5
kernel module pata_marvell SCSI low-level driver for Marvell ATA in legacy mode: 0.1.6
kernel module pata_mpiix low-level driver for Intel MPIIX: 0.7.7
kernel module pata_netcell SCSI low-level driver for Netcell PATA RAID: 0.1.7
kernel module pata_ninja32 low-level driver for Ninja32 ATA: 0.1.5
kernel module pata_ns87410 low-level driver for Nat Semi 87410: 0.4.6
kernel module pata_ns87415 ATA low-level driver for NS87415 controllers: 0.0.1
kernel module pata_oldpiix SCSI low-level driver for early PIIX series controllers: 0.5.5
kernel module pata_pdc2027x libata driver module for Promise PDC20268 to PDC20277: 1.0
kernel module pata_pdc202xx_old low-level driver for Promise 2024x and 20262-20267: 0.4.3
kernel module pata_platform low-level driver for platform device ATA: 1.2

kernel module pata_rdc SCSI low-level driver for RDC PATA controllers: 0.01
kernel module pata_rz1000 low-level driver for RZ1000 PCI ATA: 0.2.4
kernel module pata_sch SCSI low-level driver for Intel SCH PATA controllers: 0.2
kernel module pata_serverworks low-level driver for Serverworks OSB4/CSB5/CSB6: 0.4.3
kernel module pata_sil680 low-level driver for Si680 PATA: 0.4.9
kernel module pata_sis SCSI low-level driver for SiS ATA: 0.5.2
kernel module pata_sl82c105 low-level driver for Si82c105: 0.3.3
kernel module pata_triflex low-level driver for Compaq Triflex: 0.2.8
kernel module pata_via low-level driver for VIA PATA: 0.3.4
kernel module pdc_adma Pacific Digital Corporation ADMA low-level driver: 1.0
kernel module pm80xx PMC-Sierra PM8001/8006/8081/8088/8089/8074/8076/8077/8070/8072 SAS/SATA controller driver: 0.1.40
kernel module pmcraid PMC Sierra MaxRAID Controller Driver: 1.0.3
kernel module qed QLogic FastLinQ 4xxxx Core Module: 8.37.0.20
kernel module qede QLogic FastLinQ 4xxxx Ethernet Driver: 8.37.0.20
kernel module qedf QLogic FastLinQ 4xxxx FCoE Module: 8.42.3.0
kernel module qedi QLogic FastLinQ 4xxxx iSCSI Module: 8.37.0.20
kernel module qla1280 Qlogic ISP SCSI (qla1x80/qla1x160) driver: 3.27.1
kernel module qla3xxx QLogic ISP3XXX Network Driver v2.03.00-k5 : v2.03.00-k5
kernel module qla4xxx QLogic iSCSI HBA Driver: 5.04.00-k6
kernel module qlcnic QLogic 1/10 GbE Converged/Intelligent Ethernet Driver: 5.3.66
kernel module r6040 RDC R6040 NAPI PCI FastEthernet driver: 0.29 04Jul2016
kernel module rsxx IBM Flash Adapter 900GB Full Height Device Driver: 4.0.3.2516
kernel module s2io : 2.0.26.28
kernel module sata_dwc_460ex DesignWare Cores SATA controller low level driver: 1.3
kernel module sata_mv SCSI low-level driver for Marvell SATA controllers: 1.28
kernel module sata_nv low-level driver for NVIDIA nForce SATA controller: 3.5
kernel module sata_promise Promise ATA TX2/TX4/TX4000 low-level driver: 2.12
kernel module sata_qstor Pacific Digital Corporation QStor SATA low-level driver: 0.09
kernel module sata_sil low-level driver for Silicon Image SATA controller: 2.4
kernel module sata_sis low-level driver for Silicon Integrated Systems SATA controller: 1.0
kernel module sata_svw low-level driver for K2 SATA controller: 2.3
kernel module sata_sx4 Promise SATA low-level driver: 0.12
kernel module sata_uli low-level driver for ULI Electronics SATA controller: 1.3
kernel module sata_via SCSI low-level driver for VIA SATA controllers: 2.6
kernel module sata_vsc low-level driver for Vitesse VSC7174 SATA controller: 2.3
kernel module sfc_falcon Solarflare Falcon network driver: 4.1
kernel module sg SCSI generic (sg) driver: 3.5.36
kernel module sis190 SiS sis190/191 Gigabit Ethernet driver: 1.4
kernel module skge SysKonnect Gigabit Ethernet driver: 1.14
kernel module sky2 Marvell Yukon 2 Gigabit Ethernet driver: 1.30
kernel module smartpqi Driver for Microsemi Smart Family Controller version 2.1.8-045: 2.1.8-045
kernel module smsc911x : 2008-10-21
kernel module smsc9420 : 1.01
kernel module snic Cisco SCSI NIC Driver: 0.0.1.18
kernel module stex Promise Technology SuperTrak EX Controllers: 6.02.0000.01
kernel module sunhme Sun HappyMealEthernet(HME) 10/100baseT ethernet driver: 3.10
kernel module sym53c8xx NCR, Symbios and LSI 8xx and 1010 PCI SCSI adapters: 2.2.3
kernel module thunder_bgx Cavium Thunder BGX/MAC Driver: 1.0
kernel module thunder_xcv Cavium Thunder RGX/XCV Driver: 1.0
kernel module tpm TPM Driver: 2.0
kernel module tpm_atmel TPM Driver: 2.0
kernel module tpm_crb TPM2 Driver: 0.1
kernel module tpm_i2c_infineon TPM TIS I2C Infineon Driver: 2.2.0
kernel module tpm_infineon Driver for Infineon TPM SLD 9630 TT 1.1 / SLB 9635 TT 1.2: 1.9.2

kernel module tpm_nsc TPM Driver: 2.0
kernel module tpm_st33zp24 ST33ZP24 TPM 1.2 driver: 1.3.0
kernel module tpm_st33zp24_i2c STM TPM 1.2 I2C ST33 Driver: 1.3.0
kernel module tpm_tis TPM Driver: 2.0
kernel module tpm_tis_core TPM Driver: 2.0
kernel module tpm_vtpm_proxy vTPM Driver: 0.1
kernel module ufshcd_core Generic UFS host controller driver Core: 0.2
kernel module virtio_pci virtio-pci: 1
kernel module virtio_pci_modern_dev Modern Virtio PCI Device: 0.1
kernel module vmw_pvscsi VMware PVSCSI driver: 1.0.7.0-k
kernel module vmxnet3 VMware vmxnet3 virtual NIC driver: 1.5.0.0-k
kernel module vxlan Driver for VXLAN encapsulated traffic: 0.1
adduser version: 3.118
apt version: 2.2.4
apt-utils version: 2.2.4
base-files version: 11.1+deb11u2
base-passwd version: 3.5.51
bash version: 5.1-2+b3
bsdutils version: 1:2.36.1-8
ca-certificates version: 20210119
coreutils version: 8.32-4+b1
cpio version: 2.13+dfsg-4
cron version: 3.0pl1-137
cryptsetup-bin version: 2:2.3.5-1
curl version: 7.74.0-1.3+deb11u1
dash version: 0.5.11+git20200708+dd9ef66-5
debconf version: 1.5.77
debconf-i18n version: 1.5.77
debian-archive-keyring version: 2021.1.1
debiantools version: 4.11.2
dhcpcd5 version: 7.1.0-2+b1
diffutils version: 1:3.7-5
dirmngr version: 2.2.27-2
dmidecode version: 3.3-2
dmsetup version: 2:1.02.175-2.1
dosfstools version: 4.2-1
dpkg version: 1.20.9
e2fsprogs version: 1.46.2-2
ethtool version: 1:5.9-1
fdisk version: 2.36.1-8
findutils version: 4.8.0-1
fontconfig-config version: 2.13.1-4.2
fonts-dejavu-core version: 2.37-2
gcc-10-base version: 10.2.1-6
gcc-9-base version: 9.3.0-22
gdisk version: 1.0.6-1.1
gnupg version: 2.2.27-2
gnupg-l10n version: 2.2.27-2
gnupg-utils version: 2.2.27-2
gpg version: 2.2.27-2
gpg-agent version: 2.2.27-2
gpg-wks-client version: 2.2.27-2
gpg-wks-server version: 2.2.27-2
gpgconf version: 2.2.27-2

gpgsm version: 2.2.27-2
gpgv version: 2.2.27-2
grep version: 3.6-1
groff-base version: 1.22.4-6
gzip version: 1.10-4
hdparm version: 9.60+ds-1
hostname version: 3.23
hwinfo version: 21.72-1
ifenslave version: 2.12
ifupdown version: 0.8.36
init version: 1.60
init-system-helpers version: 1.60
iproute2 version: 5.10.0-4
iptables version: 1.8.7-1
iputils-ping version: 3:20210202-1
irqbalance version: 1.7.0-1
isc-dhcp-client version: 4.4.1-2.3
isc-dhcp-common version: 4.4.1-2.3
kmod version: 28-1
less version: 551-2
libacl1 version: 2.2.53-10
libapparmor1 version: 2.13.6-10
libapt-pkg6.0 version: 2.2.4
libargon2-1 version: 0~20171227-0.2
libassuan0 version: 2.5.3-7.1
libattr1 version: 1:2.4.48-6
libaudit-common version: 1:3.0-2
libaudit1 version: 1:3.0-2
libblkid1 version: 2.36.1-8
libboost-numpy1.74.0 version: 1.74.0-9
libboost-python1.74.0 version: 1.74.0-9
libboost-system1.74.0 version: 1.74.0-9
libboost-thread1.74.0 version: 1.74.0-9
libbpf0 version: 1:0.3-2
libbrotli1 version: 1.0.9-2+b2
libbsd0 version: 0.11.3-1
libbz2-1.0 version: 1.0.8-4
libc-bin version: 2.31-13+deb11u2
libc-dev-bin version: 2.31-13+deb11u2
libc-devtools version: 2.31-13+deb11u2
libc6 version: 2.31-13+deb11u2
libc6-dev version: 2.31-13+deb11u2
libcap-ng0 version: 0.7.9-2.2+b1
libcap2 version: 1:2.44-1
libcap2-bin version: 1:2.44-1
libcom-err2 version: 1.46.2-2
libcrypt-dev version: 1:4.4.18-4
libcrypt1 version: 1:4.4.18-4
libcryptsetup12 version: 2:2.3.5-1
libcurl4 version: 7.74.0-1.3+deb11u1
libdb5.3 version: 5.3.28+dfsg1-0.8
libdebconfclient0 version: 0.260
libdeflate0 version: 1.7-1
libdevmapper1.02.1 version: 2:1.02.175-2.1

libdns-export1110 version: 1:9.11.19+dfsg-2.1
libedit2 version: 3.1-20191231-2+b1
libelf1 version: 0.183-1
libestr0 version: 0.1.10-2.1+b1
libevent-core-2.1-7 version: 2.1.12-stable-1
libevent-pthreads-2.1-7 version: 2.1.12-stable-1
libexpat1 version: 2.2.10-2
libexpat1-dev version: 2.2.10-2
libext2fs2 version: 1.46.2-2
libfastjson4 version: 0.99.9-1
libfdisk1 version: 2.36.1-8
libffi7 version: 3.3-6
libfontconfig1 version: 2.13.1-4.2
libfreetype6 version: 2.10.4+dfsg-1
libgcc-s1 version: 10.2.1-6
libgcrypt20 version: 1.8.7-6
libgd3 version: 2.3.0-2
libgdbm-compat4 version: 1.19-2
libgdbm6 version: 1.19-2
libglib2.0-0 version: 2.66.8-1
libglib2.0-data version: 2.66.8-1
libgmp10 version: 2:6.2.1+dfsg-1+deb11u1
libgnutls-openssl27 version: 3.7.1-5
libgnutls30 version: 3.7.1-5
libgpg-error0 version: 1.38-2
libgpm2 version: 1.20.7-8
libgssapi-krb5-2 version: 1.18.3-6+deb11u1
libhd21 version: 21.72-1
libhogweed6 version: 3.7.3-1
libicu67 version: 67.1-7
libidn2-0 version: 2.3.0-5
libip4tc2 version: 1.8.7-1
libip6tc2 version: 1.8.7-1
libisc-export1105 version: 1:9.11.19+dfsg-2.1
libjansson4 version: 2.13.1-1.1
libjbig0 version: 2.1-3.1+b2
libjpeg62-turbo version: 1:2.0.6-4
libjson-c5 version: 0.15-2
libk5crypto3 version: 1.18.3-6+deb11u1
libkeyutils1 version: 1.6.1-2
libkmod2 version: 28-1
libkrb5-3 version: 1.18.3-6+deb11u1
libkrb5support0 version: 1.18.3-6+deb11u1
libksba8 version: 1.5.0-3
libldap-2.4-2 version: 2.4.57+dfsg-3
libldap-common version: 2.4.57+dfsg-3
liblocale-gettext-perl version: 1.07-4+b1
liblognorm5 version: 2.0.5-1.1
liblz4-1 version: 1.9.3-2
liblzma5 version: 5.2.5-2
libmd0 version: 1.0.3-3
libmnl0 version: 1.0.4-3
libmount1 version: 2.36.1-8
libmpdec3 version: 2.5.1-1

libncurses5 version: 6.2+20201114-2
libncurses6 version: 6.2+20201114-2
libncursesw6 version: 6.2+20201114-2
libnetfilter-contrack3 version: 1.0.8-3
libnettle8 version: 3.7.3-1
libnewt0.52 version: 0.52.21-4+b3
libnfnetwork0 version: 1.0.1-3+b1
libnftables1 version: 0.9.8-3.1
libnftnl11 version: 1.1.9-1
libnghttp2-14 version: 1.43.0-1
libnptl0 version: 1.6-3
libnsl-dev version: 1.3.0-2
libnsl2 version: 1.3.0-2
libnuma1 version: 2.0.12-1+b1
libopenipmi0 version: 2.0.29-0.1+b1
libopts25 version: 1:5.18.16-4
libp11-kit0 version: 0.23.22-1
libpam-modules version: 1.4.0-9+deb11u1
libpam-modules-bin version: 1.4.0-9+deb11u1
libpam-runtime version: 1.4.0-9+deb11u1
libpam0g version: 1.4.0-9+deb11u1
libpci3 version: 1:3.7.0-5
libpcre2-8-0 version: 10.36-2
libpcre3 version: 2:8.39-13
libperl5.32 version: 5.32.1-4+deb11u2
libpng16-16 version: 1.6.37-3
libpopt0 version: 1.18-2
libprocps8 version: 2:3.3.17-5
libpsl5 version: 0.21.0-1.2
libpython3.9 version: 3.9.2-1
libpython3.9-dev version: 3.9.2-1
libpython3.9-minimal version: 3.9.2-1
libpython3.9-stdlib version: 3.9.2-1
libreadline8 version: 8.1-1
librtmp1 version: 2.4+20151223.gitfa8646d.1-2+b2
libsasldb2 version: 2.1.27+dfsg-2.1
libsasldb-modules version: 2.1.27+dfsg-2.1
libsasldb-modules-db version: 2.1.27+dfsg-2.1
libseccomp2 version: 2.5.1-1+deb11u1
libselinux1 version: 3.1-3
libsemanage-common version: 3.1-1
libsemanage1 version: 3.1-1+b2
libsensors-config version: 1:3.6.0-7
libsensors5 version: 1:3.6.0-7
libsepol1 version: 3.1-1
libsgutils2-2 version: 1.45-1
libslang2 version: 2.3.2-5
libsmartcols1 version: 2.36.1-8
libsnmp-base version: 5.9+dfsg-3.1
libsnmp40 version: 5.9+dfsg-3.1
libsqlite3-0 version: 3.34.1-3
libss2 version: 1.46.2-2
libssh2-1 version: 1.9.0-2
libssl1.1 version: 1.1.1k-1+deb11u1

libstdc++6 version: 10.2.1-6
libsfs2 version: 2.1.0+repack-7
libsystemd0 version: 247.3-6
libtasn1-6 version: 4.16.0-2
libtext-charwidth-perl version: 0.04-10+b1
libtext-iconv-perl version: 1.7-7+b1
libtext-wrapi18n-perl version: 0.06-9
libtiff5 version: 4.2.0-1
libtinfo5 version: 6.2+20201114-2
libtinfo6 version: 6.2+20201114-2
libtirpc-common version: 1.3.1-1
libtirpc-dev version: 1.3.1-1
libtirpc3 version: 1.3.1-1
libuchardet0 version: 0.0.7-1
libudev1 version: 247.3-6
libunistring2 version: 0.9.10-4
liburing1 version: 0.7-3
libuuid1 version: 2.36.1-8
libwebp6 version: 0.6.1-2.1
libwrap0 version: 7.6.q-31
libx11-6 version: 2:1.7.2-1
libx11-data version: 2:1.7.2-1
libx86emu3 version: 3.1-2
libxau6 version: 1:1.0.9-1
libxcb1 version: 1.14-3
libxdmcp6 version: 1:1.1.2-3
libxml2 version: 2.9.10+dfsg-6.7
libxpm4 version: 1:3.5.12-1
libxtables12 version: 1.8.7-1
libxxhash0 version: 0.8.0-2
libyajl2 version: 2.1.0-3
libzstd1 version: 1.4.8+dfsg-2.1
linux-firmware version: 1.201
linux-libc-dev version: 5.10.84-1
login version: 1:4.8.1-1
logrotate version: 3.18.0-2
logsave version: 1.46.2-2
lsb-base version: 11.1.0
lshw version: 02.19.git.2021.06.19.996aaad9c7-2~bpo11+1
lsscsi version: 0.31-1+b1
manpages version: 5.10-1
manpages-dev version: 5.10-1
mawk version: 1.3.4.20200120-2
media-types version: 4.0.0
mount version: 2.36.1-8
nano version: 5.4-2
ncurses-base version: 6.2+20201114-2
ncurses-bin version: 6.2+20201114-2
net-tools version: 1.60+git20181103.0eebece-1
netbase version: 6.3
nftables version: 0.9.8-3.1
ntp version: 1:4.2.8p15+dfsg-1
openipmi version: 2.0.29-0.1+b1
openresolv version: 3.12.0-1

openssl version: 1.1.1k-1+deb11u1
passwd version: 1:4.8.1-1
pci.ids version: 0.0~2021.02.08-1
perl version: 5.32.1-4+deb11u2
perl-base version: 5.32.1-4+deb11u2
perl-modules-5.32 version: 5.32.1-4+deb11u2
pinentry-curses version: 1.1.0-4
powermgmt-base version: 1.36
procps version: 2:3.3.17-5
publicsuffix version: 20211207.1025-0+deb11u1
python3.9 version: 3.9.2-1
python3.9-minimal version: 3.9.2-1
qemu-guest-agent version: 1:5.2+dfsg-11+deb11u1
readline-common version: 8.1-1
rsyslog version: 8.2102.0-2
runit-helper version: 2.10.3
sdparm version: 1.10-1+b1
sed version: 4.7-1
sensible-utils version: 0.0.14
sg3-utils version: 1.45-1
shared-mime-info version: 2.0-1
smartmontools version: 7.2-1
smp-utils version: 0.99-1
snmp version: 5.9+dfsg-3.1
snmpd version: 5.9+dfsg-3.1
sntp version: 1:4.2.8p15+dfsg-1
ssmtp version: 2.64-10
sysstat version: 12.5.2-2
systemd version: 247.3-6
systemd-sysv version: 247.3-6
systemd-timesyncd version: 247.3-6
sysvinit-utils version: 2.96-7
tar version: 1.34+dfsg-1
tasksel version: 3.68
tasksel-data version: 3.68
tofrodos version: 1.7.13+ds-5
traceroute version: 1:2.1.0-2+b1
tzdata version: 2021a-1+deb11u2
ucf version: 3.0043
udev version: 247.3-6
usb.ids version: 2021.06.06-1
util-linux version: 2.36.1-8
vim-common version: 2:8.2.2434-3+deb11u1
vim-tiny version: 2:8.2.2434-3+deb11u1
whiptail version: 0.52.21-4+b3
xdg-user-dirs version: 0.17-2
xxd version: 2:8.2.2434-3+deb11u1
xz-utils version: 5.2.5-2
zlib1g version: 1:1.2.11.dfsg-2
python package zope.interface: 5.4.0
python package ipaddress version: 1.0.23
python package cryptography version: 3.4.7
python package pyOpenSSL version: 20.0.1
python package service_identity version: 18.1.0

python package requests version: 2.25.1
python package incremental version: 21.3.0
python package Twisted[tls] version: 21.2.0
python package pyutil version: 3.3.0
python package python-dateutil version: 2.8.1
python package Werkzeug version: 1.0.1
python package klein version: 20.6.0
python package zfec version: 1.5.5
python package yajl-py version: 2.1.2
python package certifi
python package pyratemp version: 0.3.2
python package numpy version: 1.20.2

Swarm Storage 14.0.1 Release

- [New Features](#)
- [Additional Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Known Issues](#)

New Features

Erasure Coding Improvements – Swarm 14 includes Erasure Coding related improvements:

- With Swarm 14.0 indexed erasure coded (EC) objects include the field "ec_encoding" which records the current EC coding of the object. Non-EC objects do not have this field. (SWAR-6653)
- Swarm now computes the data footprint of Erasure Coding EC segments and whole replicas objects separately during each HP cycle so the relative space usage of whole replicas vs EC can inform space usage policy decisions. (SWAR-9160)

Additional Changes

These items are other changes, including those that come from testing and user feedback.

- **OSS Versions** – See [Third-Party Components for Storage 14.0.1](#) for the complete listing of packages and versions for this release.
- **Fixed in 14.0**
 - **Reboot loop due to a bad disk:** The Swarm node recognizes the volume as failed and alerts the cluster to the failure when a volume fails at mount time. The node operates with the remaining volumes, so physically removing the volume may be necessary. (SWAR-9189)
 - **Remove legacy nonce handling:** Remove `scsp.forceLegacyNonce` settings from the `node.cfg` files prior to upgrading to 14.0. (SWAR-9108)
 - **Bucket listings:** During a node reboot, such as a rolling reboot of the cluster, a newly booted node temporarily returns an empty result set for a listing query. (SWAR-9083)
 - **S3 backup feed:** A 5G object size limitation is removed. (SWAR-8554)

Upgrade Impacts



Required

Complete the migration to Swarm 11.3 and ES 6.8.6 before upgrading to Swarm 14 if running older Elasticsearch (5.6.12 or 2.3.3). See [here](#), *Upgrading from Unsupported Elasticsearch*.

These items are changes to the product function that may require operational or development changes for integrated applications. Address the upgrade impacts for each of the versions since the currently running version:

Impacts for 14.0

- **Change in the `node.cfg` file** – The previously deprecated `sysctl` section of the `node.cfg` file is removed. Use `kernel.sysctlFileUrl` (introduced in Swarm 12.0) instead if it is necessary to set kernel runtime parameters, . (SWAR-8968)

Settings changes

- Updated:
 - All `sysctl.*` settings are removed. (SWAR-8968)
 - `support.reportPeriod` default is changed to 21600 (6 hours). (SWAR-8424)
 - The following settings are now persisted cluster settings that can be updated via SNMP and the UI (SWAR-9115)
 - `cluster.enforceTenancy`
 - `cluster.proxyIPList`
 - `ec.maxManifests`
 - `ec.minParity`
 - `ec.segmentSize`
 - `feeds.retry`
 - `health.parallelWriteTimeout`
 - `health.underreplicationAlertPercent`
 - `health.underreplicationTolerance`
 - `health.persistentUnderreplicationAlertPercent`
 - `log.obscureUUIDs`
 - `scsp.clientPoolTimeout`
 - `scsp.defaultContextReplicas`
 - `scsp.defaultROWAction`
 - `scsp.maxWriteTime`
 - `scsp.validateOnRead`
 - `search.numberOfShards`
- **Swarm storage node metrics are deprecated** and are replaced in the next major release by the graphs and reporting from [Grafana and Prometheus Node Exporter](#). The storage administration UI is updated to allow for metrics to be turned off. Clear `metrics.target` from the configuration, uninstall `caringo-elasticsearch-metrics`, and `curl -XDELETE 'http://ELASTICSEARCH:9200/metrics-*` to clear the space in the Elasticsearch cluster. (SWAR-8982)
- **Differences in `scsp.forceLegacyNonce` configuration** depending on the version upgrading from (SWAR-9020):
- **Currently running a Swarm Storage version prior to 11.1**, and upgrading to 11.1, 11.2, 11.3, 12.0 or 12.1:

Before upgrading, set `scsp.forceLegacyNonce=true` in the `node.cfg` file. After the upgrade, when the cluster is fully up, update `scsp.forceLegacyNonce=false` using `swarmctl` and change `scsp.forceLegacyNonce=false` in the `node.cfg` file.

Currently running a Swarm Storage version 11.1, 11.2, 11.3, 12.0 or 12.1 and upgrading to another version from that list:

Before upgrading, verify `scsp.forceLegacyNonce=false` is in the `node.cfg` file and verify using `swarmctl` that `scsp.forceLegacyNonce=false` in a cluster.

- **Subcluster assignments can no longer be blank**, and CSN installations with mixed subcluster assignments have the unassigned nodes unable to boot, showing an error in contacting the time source. Supply a subcluster for each node if any named subcluster is specified in a cluster. (SWAR-7675)

Use `swarmctl` to check or change settings

Use `'swarmctl -C scsp.forceLegacyNonce'` to check the value of `scsp.forceLegacyNonce`.

Use `'swarmctl -C scsp.forceLegacyNonce -V False'` to set the value to false.

For more details, see <https://support.cloud.caringo.com/tools/Tech-Support-Scripts-Bundle-swarmctl.pdf>.



Cumulative impacts

Address all upgrade impacts for *each version released* since the version being upgraded from.

Review the comprehensive *Upgrade Impacts* listed for the [Swarm Storage 14 Release](#).

Watch Items and Known Issues

The following watch items are known:

- A node fails to mount all disks in the node if a node mounts an encrypted volume that is missing the encryption key in the configuration. (SWAR-8762)
- S3 Backup feeds do not backup logical objects greater than 5 GB; those writes fail with a **CRITICAL** log message. (SWAR-8554)
- The chassis shuts down but does not come back up when restarting a cluster of virtual machines that are UEFI-booted (versus legacy BIOS). (SWAR-8054)
- With multipath-enabled hardware, the Swarm console Disk Volume Menu may erroneously show too many disks, having multiplied the actual disks in use by the number of possible paths to them. (SWAR-7248)

These are standing operational limitations:

- The Storage UI shows no NFS config if the Elasticsearch cluster is wiped. Contact DataCore Support for help repopulating the SwarmFS config information. (SWAR-8007)
- Any incomplete multipart upload into a bucket leaves the parts (unnamed streams) in the domain if a bucket is deleted. To find and delete them, use the `s3cmd` utility (search the Support site for "`s3cmd`" for guidance). (SWAR-7690)
- Invalid config parameters that prevent the unassigned nodes from booting are created if subcluster assignments are removed in the CSN UI. (SWAR-7675)
- False 404 Not Found and other SCSP errors may display during rolling reboot in versions 11.1 through 12.0.1. Set `scsp.forceLegacyNonce=False` in the cluster configuration to mitigate this problem. This setting needs to be removed before upgrading to 12.1.0 or later. (SWAR-9020)
- S3 Backup restoration to the cluster may be blocked if the certificate is not located where Swarm expects it when using certificates with HAProxy. From 12.1, a clearer error message draws attention to the issue for S3 backup and replication feeds that are blocked due to invalid X.509 ("SSL") certificates. (SWAR-8996)

To upgrade Swarm 9 or higher, proceed now to [How to Upgrade Swarm](#). For migration from Swarm 8.x or earlier, contact DataCore Support for guidance.

Third-Party Components for Storage 14.0.1

For licensing information, see [Open Source Software Licenses](#).

Elasticsearch and Swarm distributions

Elasticsearch 7.5.2
 Swarm S3 Backup Restore 1.2.4
 Swarm Search 7.0.1
 Swarm Metrics 7.0.1

Swarm Storage components

Operating system: Debian GNU/Linux 10 (buster)
 Linux kernel: 5.4.109
 kernel module 3w_9xxx 3ware 9000 Storage Controller Linux Driver: 2.26.02.014
 kernel module 3w_sas LSI 3ware SAS/SATA-RAID Linux Driver: 3.26.02.000
 kernel module 3w_xxxx 3ware Storage Controller Linux Driver: 1.26.02.003
 kernel module 8021q : 1.8
 kernel module 8139cp RealTek RTL-8139C+ series 10/100 PCI Ethernet driver: 1.3
 kernel module 8139too RealTek RTL-8139 Fast Ethernet driver: 0.9.28
 kernel module aacraid Dell PERC2, 2/Si, 3/Si, 3/Di, Adaptec Advanced Raid Products, HP NetRAID-4M, IBM ServeRAID & ICP SCSI driver: 1.2.1
 [50877]-custom
 kernel module acard_ahci ACard AHCI SATA low-level driver: 1.0
 kernel module ad7418 AD7416/17/18 driver: 0.4
 kernel module ahci AHCI SATA low-level driver: 3.0
 kernel module aic79xx Adaptec AIC790X U320 SCSI Host Bus Adapter driver: 3.0
 kernel module aic7xxx Adaptec AIC77XX/78XX SCSI Host Bus Adapter driver: 7.0
 kernel module aic94xx Adaptec aic94xx SAS/SATA driver: 1.0.3
 kernel module am53c974 AM53C974 SCSI driver: 1.00
 kernel module amd_xgbe AMD 10 Gigabit Ethernet Driver: 1.0.3
 kernel module arcmsr Areca ARC11xx/12xx/16xx/18xx SAS/SATA RAID Controller Driver: v1.40.00.10-20190116
 kernel module ata_generic low-level driver for generic ATA: 0.2.15
 kernel module ata_piix SCSI low-level driver for Intel PIIX/ICH ATA controllers: 2.13
 kernel module atl1 Atheros L1 Gigabit Ethernet Driver: 2.1.3
 kernel module atl1c Qualcomm Atheros 100/1000M Ethernet Network Driver: 1.0.1.1-NAPI
 kernel module atl1e Atheros 1000M Ethernet Network Driver: 1.0.0.7-NAPI
 kernel module atl2 Atheros Fast Ethernet Network Driver: 2.2.3
 kernel module atlantic aQuantia Corporation(R) Network Driver: 5.4.109-kern
 kernel module atxp1 System voltages control via Attansic ATXP1: 0.6.3
 kernel module b44 Broadcom 44xx/47xx 10/100 PCI ethernet driver: 2.0
 kernel module be2iscsi Emulex OneConnectOpen-iSCSI Driver version11.4.0.1 Driver 11.4.0.1: 11.4.0.1
 kernel module be2net Emulex OneConnect NIC Driver 12.0.0.0: 12.0.0.0
 kernel module bfa QLogic BR-series Fibre Channel HBA Driver fcpim: 3.2.25.1
 kernel module bna QLogic BR-series 10G PCIe Ethernet driver: 3.2.25.1
 kernel module bnx2 QLogic BCM5706/5708/5709/5716 Driver: 2.2.6
 kernel module bnx2fc QLogic FCoE Driver: 2.12.10
 kernel module bnx2i QLogic NetXtreme II BCM5706/5708/5709/57710/57711/57712/57800/57810/57840 iSCSI Driver: 2.7.10.1
 kernel module bnx2x QLogic BCM57710/57711/57711E/57712/57712_MF/57800/57800_MF/57810/57810_MF/57840/57840_MF Driver: 1.713.36-0
 kernel module bnxt_en Broadcom BCM573xx network driver: 1.10.0

kernel module bonding Ethernet Channel Bonding Driver, v3.7.1: 3.7.1
kernel module cnic QLogic cnic Driver: 2.5.22
kernel module csiostor Chelsio FCoE driver: 1.0.0-ko
kernel module cxgb3 Chelsio T3 Network Driver: 1.1.5-ko
kernel module cxgb3i Chelsio T3 iSCSI Driver: 2.0.1-ko
kernel module cxgb4 Chelsio T4/T5/T6 Network Driver: 2.0.0-ko
kernel module cxgb4i Chelsio T4-T6 iSCSI Driver: 0.9.5-ko
kernel module cxgb4vf Chelsio T4/T5/T6 Virtual Function (VF) Network Driver: 2.0.0-ko
kernel module dca : 1.12.1
kernel module dcdbas Dell Systems Management Base Driver (version 5.6.0-3.3): 5.6.0-3.3
kernel module de2104x Intel/Digital 21040/1 series PCI Ethernet driver: 0.7
kernel module dmfe Davicom DM910X fast ethernet driver: 1.36.4
kernel module e100 Intel(R) PRO/100 Network Driver: 3.5.24-k2-NAPI
kernel module e1000 Intel(R) PRO/1000 Network Driver: 7.3.21-k8-NAPI
kernel module e1000e Intel(R) PRO/1000 Network Driver: 3.2.6-k
kernel module eeprom_93cx6 EEPROM 93cx6 chip driver: 1.0
kernel module efivars sysfs interface to EFI Variables: 0.08
kernel module ena Elastic Network Adapter (ENA): 2.1.0K
kernel module enic Cisco VIC Ethernet NIC Driver: 2.3.0.53
kernel module esas2r esas2r: 1.00
kernel module esp_scsi ESP SCSI driver core: 2.000
kernel module fm10k Intel(R) Ethernet Switch Host Interface Driver: 0.26.1-k
kernel module fnic Cisco FCoE HBA Driver: 1.6.0.47
kernel module hpsa Driver for HP Smart Array Controller version 3.4.20-170: 3.4.20-170
kernel module i40e Intel(R) 40-10 Gigabit Ethernet Connection Network Driver: 2.10.19.82
kernel module i40e Intel(R) Ethernet Connection XL710 Network Driver: 2.8.20-k
kernel module iavf Intel(R) Ethernet Adaptive Virtual Function Network Driver: 3.2.3-k
kernel module ice Intel(R) Ethernet Connection E800 Series Linux Driver: 0.8.1-k
kernel module igb Intel(R) Gigabit Ethernet Network Driver: 5.6.0-k
kernel module igbvf Intel(R) Gigabit Virtual Function Network Driver: 2.4.0-k
kernel module ioatdma : 5.00
kernel module ipmi_msghandler Incoming and outgoing message routing for an IPMI interface.: 39.2
kernel module ipr IBM Power RAID SCSI Adapter Driver: 2.6.4
kernel module ips IBM ServerAID Adapter Driver 7.12.05: 7.12.05
kernel module iscsi : 1.2.0
kernel module ixgb Intel(R) PRO/10GbE Network Driver: 1.0.135-k2-NAPI
kernel module ixgbe Intel(R) 10 Gigabit PCI Express Network Driver: 5.1.0-k
kernel module ixgbe Intel(R) 10GbE PCI Express Linux Network Driver: 5.11.3
kernel module ixgbevfn Intel(R) 10 Gigabit Virtual Function Network Driver: 4.1.0-k
kernel module jme JMicron JMC2x0 PCI Express Ethernet driver: 1.0.8
kernel module libcxgb Chelsio common library: 1.0.0-ko
kernel module libcxgbi Chelsio iSCSI driver library: 0.9.1-ko
kernel module liquidio Cavium LiquidIO Intelligent Server Adapter Driver: 1.7.2
kernel module liquidio_vf Cavium LiquidIO Intelligent Server Adapter Virtual Function Driver: 1.7.2
kernel module lpfc Emulex LightPulse Fibre Channel SCSI driver 12.4.0.0: 0
kernel module megaraid LSI Logic MegaRAID legacy driver: 2.00.4
kernel module megaraid_mbox LSI Logic MegaRAID Mailbox Driver: 2.20.5.1
kernel module megaraid_mm LSI Logic Management Module: 2.20.2.7
kernel module megaraid_sas Broadcom MegaRAID SAS Driver: 07.710.50.00-rc1
kernel module mlx4_core Mellanox ConnectX HCA low-level driver: 4.0-0
kernel module mlx4_en Mellanox ConnectX HCA Ethernet driver: 4.0-0
kernel module mlx5_core Mellanox 5th generation network adapters (ConnectX series) core driver: 5.0-0
kernel module mpt3sas LSI MPT Fusion SAS 3.0 Device Driver: 31.100.00.00
kernel module mptbase Fusion MPT base driver: 3.04.20

kernel module mptctl Fusion MPT misc device (ioctl) driver: 3.04.20
 kernel module mptfc Fusion MPT FC Host driver: 3.04.20
 kernel module mptsas Fusion MPT SAS Host driver: 3.04.20
 kernel module mptscsih Fusion MPT SCSI Host driver: 3.04.20
 kernel module mptspi Fusion MPT SPI Host driver: 3.04.20
 kernel module mtip32xx Micron RealSSD PCIe Block Driver: 1.3.1
 kernel module mvsas Marvell 88SE6440 SAS/SATA controller driver: 0.8.16
 kernel module myri10ge Myricom 10G driver (10GbE): 1.5.3-1.534
 kernel module netxen_nic QLogic/NetXen (1/10) GbE Intelligent Ethernet Driver: 4.0.82
 kernel module nfp The Netronome Flow Processor (NFP) driver.: 5.4.109
 kernel module nicpf Cavium Thunder NIC Physical Function Driver: 1.0
 kernel module nicvf Cavium Thunder NIC Virtual Function Driver: 1.0
 kernel module niu NIU ethernet driver: 1.1
 kernel module nvme : 1.0
 kernel module nvme_core : 1.0
 kernel module pata_acpi SCSI low-level driver for ATA in ACPI mode: 0.2.3
 kernel module pata_ali low-level driver for ALi PATA: 0.7.8
 kernel module pata_amd low-level driver for AMD and Nvidia PATA IDE: 0.4.1
 kernel module pata_artop SCSI low-level driver for ARTOP PATA: 0.4.6
 kernel module pata_atiixp low-level driver for ATI IXP200/300/400: 0.4.6
 kernel module pata_atp867x low level driver for Artop/Acard 867x ATA controller: 0.7.5
 kernel module pata_cmd64x low-level driver for CMD64x series PATA controllers: 0.2.18
 kernel module pata_efar SCSI low-level driver for EFAR PIIX clones: 0.4.5
 kernel module pata_hpt366 low-level driver for the Highpoint HPT366/368: 0.6.11
 kernel module pata_hpt37x low-level driver for the Highpoint HPT37x/30x: 0.6.23
 kernel module pata_hpt3x2n low-level driver for the Highpoint HPT3xxN: 0.3.15
 kernel module pata_hpt3x3 low-level driver for the Highpoint HPT343/363: 0.6.1
 kernel module pata_it821x low-level driver for the IT8211/IT8212 IDE RAID controller: 0.4.2
 kernel module pata_jmicron SCSI low-level driver for Jmicron PATA ports: 0.1.5
 kernel module pata_marvell SCSI low-level driver for Marvell ATA in legacy mode: 0.1.6
 kernel module pata_mpiix low-level driver for Intel MPIIX: 0.7.7
 kernel module pata_netcell SCSI low-level driver for Netcell PATA RAID: 0.1.7
 kernel module pata_ninja32 low-level driver for Ninja32 ATA: 0.1.5
 kernel module pata_ns87410 low-level driver for Nat Semi 87410: 0.4.6
 kernel module pata_ns87415 ATA low-level driver for NS87415 controllers: 0.0.1
 kernel module pata_oldpiix SCSI low-level driver for early PIIX series controllers: 0.5.5
 kernel module pata_pdc2027x libata driver module for Promise PDC20268 to PDC20277: 1.0
 kernel module pata_pdc202xx_old low-level driver for Promise 2024x and 20262-20267: 0.4.3
 kernel module pata_platform low-level driver for platform device ATA: 1.2
 kernel module pata_rdc SCSI low-level driver for RDC PATA controllers: 0.01
 kernel module pata_rz1000 low-level driver for RZ1000 PCI ATA: 0.2.4
 kernel module pata_sch SCSI low-level driver for Intel SCH PATA controllers: 0.2
 kernel module pata_serverworks low-level driver for Serverworks OSB4/CSB5/CSB6: 0.4.3
 kernel module pata_sil680 low-level driver for SI680 PATA: 0.4.9
 kernel module pata_sis SCSI low-level driver for SiS ATA: 0.5.2
 kernel module pata_sl82c105 low-level driver for SI82c105: 0.3.3
 kernel module pata_triflex low-level driver for Compaq Triflex: 0.2.8
 kernel module pata_via low-level driver for VIA PATA: 0.3.4
 kernel module pdc_adma Pacific Digital Corporation ADMA low-level driver: 1.0
 kernel module pm80xx PMC-Sierra PM8001/8006/8081/8088/8089/8074/8076/8077/8070/8072 SAS/SATA controller driver: 0.1.39
 kernel module pmcraid PMC Sierra MaxRAID Controller Driver: 1.0.3
 kernel module qed QLogic FastLinQ 4xxxx Core Module: 8.37.0.20
 kernel module qede QLogic FastLinQ 4xxxx Ethernet Driver: 8.37.0.20
 kernel module qedf QLogic FastLinQ 4xxxx FCoE Module: 8.42.3.0

kernel module qedi QLogic FastLinQ 4xxx iSCSI Module: 8.37.0.20
 kernel module qla1280 Qlogic ISP SCSI (qla1x80/qla1x160) driver: 3.27.1
 kernel module qla2xxx QLogic Fibre Channel HBA Driver: 10.01.00.19-k
 kernel module qla3xxx QLogic ISP3XXX Network Driver v2.03.00-k5 : v2.03.00-k5
 kernel module qla4xxx QLogic iSCSI HBA Driver: 5.04.00-k6
 kernel module qlcnic QLogic 1/10 GbE Converged/Intelligent Ethernet Driver: 5.3.66
 kernel module r6040 RDC R6040 NAPI PCI FastEthernet driver: 0.29 04Jul2016
 kernel module rsxx IBM Flash Adapter 900GB Full Height Device Driver: 4.0.3.2516
 kernel module s2io : 2.0.26.28
 kernel module sata_dwc_460ex DesignWare Cores SATA controller low level driver: 1.3
 kernel module sata_mv SCSI low-level driver for Marvell SATA controllers: 1.28
 kernel module sata_nv low-level driver for NVIDIA nForce SATA controller: 3.5
 kernel module sata_promise Promise ATA TX2/TX4/TX4000 low-level driver: 2.12
 kernel module sata_qstor Pacific Digital Corporation QStor SATA low-level driver: 0.09
 kernel module sata_sil low-level driver for Silicon Image SATA controller: 2.4
 kernel module sata_sis low-level driver for Silicon Integrated Systems SATA controller: 1.0
 kernel module sata_svw low-level driver for K2 SATA controller: 2.3
 kernel module sata_sx4 Promise SATA low-level driver: 0.12
 kernel module sata_uli low-level driver for ULi Electronics SATA controller: 1.3
 kernel module sata_via SCSI low-level driver for VIA SATA controllers: 2.6
 kernel module sata_vsc low-level driver for Vitesse VSC7174 SATA controller: 2.3
 kernel module sfc Solarflare network driver: 4.1
 kernel module sfc_falcon Solarflare Falcon network driver: 4.1
 kernel module sg SCSI generic (sg) driver: 3.5.36
 kernel module sis190 SiS sis190/191 Gigabit Ethernet driver: 1.4
 kernel module skge SysKonnect Gigabit Ethernet driver: 1.14
 kernel module sky2 Marvell Yukon 2 Gigabit Ethernet driver: 1.30
 kernel module slicoss Alacritech non-accelerated SLIC driver: 1.0
 kernel module smartpqi Driver for Microsemi Smart Family Controller version 1.2.8-026: 1.2.8-026
 kernel module smsc911x : 2008-10-21
 kernel module smsc9420 : 1.01
 kernel module snic Cisco SCSI NIC Driver: 0.0.1.18
 kernel module starfire Adaptec Starfire Ethernet driver: 2.1
 kernel module stex Promise Technology SuperTrak EX Controllers: 6.02.0000.01
 kernel module sunhme Sun HappyMealEthernet(HME) 10/100baseT ethernet driver: 3.10
 kernel module sym53c8xx NCR, Symbios and LSI 8xx and 1010 PCI SCSI adapters: 2.2.3
 kernel module tg3 Broadcom Tigon3 ethernet driver: 3.137
 kernel module thunder_bgx Cavium Thunder BGX/MAC Driver: 1.0
 kernel module thunder_xcv Cavium Thunder RGX/XCV Driver: 1.0
 kernel module tpm TPM Driver: 2.0
 kernel module tpm_atmel TPM Driver: 2.0
 kernel module tpm_crb TPM2 Driver: 0.1
 kernel module tpm_i2c_infineon TPM TIS I2C Infineon Driver: 2.2.0
 kernel module tpm_infineon Driver for Infineon TPM SLD 9630 TT 1.1 / SLB 9635 TT 1.2: 1.9.2
 kernel module tpm_nsc TPM Driver: 2.0
 kernel module tpm_st33zp24 ST33ZP24 TPM 1.2 driver: 1.3.0
 kernel module tpm_st33zp24_i2c STM TPM 1.2 I2C ST33 Driver: 1.3.0
 kernel module tpm_tis TPM Driver: 2.0
 kernel module tpm_tis_core TPM Driver: 2.0
 kernel module tpm_vtpm_proxy vTPM Driver: 0.1
 kernel module tulip Digital 21*4* Tulip ethernet driver: 1.1.15
 kernel module typhoon 3Com Typhoon Family (3C990, 3CR990, and variants): 1.0
 kernel module ufshcd_core Generic UFS host controller driver Core: 0.2
 kernel module virtio_pci virtio-pci: 1

kernel module vmw_pvscsi VMware PVSCSI driver: 1.0.7.0-k
kernel module vmxnet3 VMware vmxnet3 virtual NIC driver: 1.4.17.0-k
kernel module vxlan Driver for VXLAN encapsulated traffic: 0.1
kernel module winbond_840 Winbond W89c840 Ethernet driver: 1.01-e
adduser version: 3.118
apt version: 1.8.2.2
apt-utils version: 1.8.2.2
base-files version: 10.3+deb10u9
base-passwd version: 3.5.46
bash version: 5.0-4
bsdmainutils version: 11.1.2+b1
bsdutils version: 1:2.33.1-0.1
busybox version: 1:1.30.1-4
bzip2 version: 1.0.6-9.2~deb10u1
ca-certificates version: 20200601~deb10u2
coreutils version: 8.30-3
cpio version: 2.12+dfsg-9
cron version: 3.0pl1-134+deb10u1
cryptsetup-bin version: 2:2.1.0-5+deb10u2
curl version: 7.64.0-4+deb10u1
dash version: 0.5.10.2-5
debconf version: 1.5.71
debconf-i18n version: 1.5.71
debian-archive-keyring version: 2019.1+deb10u1
debianutils version: 4.8.6.1
dhcpcd5 version: 7.1.0-2
diffutils version: 1:3.7-3
dirmngr version: 2.2.12-1+deb10u1
dmidecode version: 3.2-1
dmsetup version: 2:1.02.155-3
dosfstools version: 4.1-2
dpkg version: 1.19.7
e2fsprogs version: 1.44.5-1+deb10u3
ethtool version: 1:4.19-1
fdisk version: 2.33.1-0.1
file version: 1:5.35-4+deb10u2
findutils version: 4.6.0+git+20190209-2
gcc-8-base version: 8.3.0-6
gdbm-l10n version: 1.18.1-4
gdisk version: 1.0.3-1.1
gnupg version: 2.2.12-1+deb10u1
gnupg-l10n version: 2.2.12-1+deb10u1
gnupg-utils version: 2.2.12-1+deb10u1
gpg version: 2.2.12-1+deb10u1
gpg-agent version: 2.2.12-1+deb10u1
gpg-wks-client version: 2.2.12-1+deb10u1
gpg-wks-server version: 2.2.12-1+deb10u1
gpgconf version: 2.2.12-1+deb10u1
gpgsm version: 2.2.12-1+deb10u1
gpgv version: 2.2.12-1+deb10u1
grep version: 3.3-1
groff-base version: 1.22.4-3+deb10u1
guile-2.2-libs version: 2.2.4+1-2+deb10u1
gzip version: 1.9-3

hdparm version: 9.58+ds-1
hostname version: 3.21
hwnfo version: 21.63-3
ifenslave version: 2.9
ifupdown version: 0.8.35
init version: 1.56+nmu1
init-system-helpers version: 1.56+nmu1
initramfs-tools version: 0.133+deb10u1
initramfs-tools-core version: 0.133+deb10u1
iproute2 version: 4.20.0-2+deb10u1
iptables version: 1.8.2-4
iputils-ping version: 3:20180629-2+deb10u2
irqbalance version: 1.5.0-3
isc-dhcp-client version: 4.4.1-2
isc-dhcp-common version: 4.4.1-2
klibc-utils version: 2.0.6-1
kmod version: 26-1
kpartx version: 0.7.9-3+deb10u1
krb5-locales version: 1.17-3+deb10u1
less version: 487-0.1+b1
libacl1 version: 2.2.53-4
libaio1 version: 0.3.112-3
libapparmor1 version: 2.13.2-10
libapt-inst2.0 version: 1.8.2.2
libapt-pkg5.0 version: 1.8.2.2
libargon2-1 version: 0~20171227-0.2
libassuan0 version: 2.5.2-1
libattr1 version: 1:2.4.48-4
libaudit-common version: 1:2.8.4-3
libaudit1 version: 1:2.8.4-3
libblkid1 version: 2.33.1-0.1
libboost-atomic1.67.0 version: 1.67.0-13+deb10u1
libboost-numpy1.67.0 version: 1.67.0-13+deb10u1
libboost-python1.67.0 version: 1.67.0-13+deb10u1
libboost-system1.67.0 version: 1.67.0-13+deb10u1
libboost-thread1.67.0 version: 1.67.0-13+deb10u1
libbsd0 version: 0.9.1-2+deb10u1
libbz2-1.0 version: 1.0.6-9.2~deb10u1
libc-bin version: 2.28-10
libc6 version: 2.28-10
libcap-ng0 version: 0.7.9-2
libcap2 version: 1:2.25-2
libcap2-bin version: 1:2.25-2
libcom-err2 version: 1.44.5-1+deb10u3
libcryptsetup12 version: 2:2.1.0-5+deb10u2
libcurl4 version: 7.64.0-4+deb10u1
libdb5.3 version: 5.3.28+dfsg1-0.5
libdebconfclient0 version: 0.249
libdevmapper1.02.1 version: 2:1.02.155-3
libdns-export1104 version: 1:9.11.5.P4+dfsg-5.1+deb10u3
libedit2 version: 3.1-20181209-1
libelf1 version: 0.176-1.1
libestr0 version: 0.1.10-2.1
libevent-core-2.1-6 version: 2.1.8-stable-4

libevent-pthreads-2.1-6 version: 2.1.8-stable-4
libexpat1 version: 2.2.6-2+deb10u1
libext2fs2 version: 1.44.5-1+deb10u3
libfastjson4 version: 0.99.8-2
libfdisk1 version: 2.33.1-0.1
libffi6 version: 3.2.1-9
libfribidi0 version: 1.0.5-3.1+deb10u1
libgc1c2 version: 1:7.6.4-0.4
libgcc1 version: 1:8.3.0-6
libgcrypt20 version: 1.8.4-5
libgdbm-compat4 version: 1.18.1-4
libgdbm6 version: 1.18.1-4
libglib2.0-0 version: 2.58.3-2+deb10u2
libglib2.0-data version: 2.58.3-2+deb10u2
libgmp10 version: 2:6.1.2+dfsg-4
libgnutls-openssl27 version: 3.6.7-4+deb10u6
libgnutls30 version: 3.6.7-4+deb10u6
libgpg-error0 version: 1.35-1
libgsasl7 version: 1.8.0-8+b2
libgssapi-krb5-2 version: 1.17-3+deb10u1
libhd21 version: 21.63-3
libhogweed4 version: 3.4.1-1
libicu63 version: 63.1-6+deb10u1
libidn11 version: 1.33-2.2
libidn2-0 version: 2.0.5-1+deb10u1
libip4tc0 version: 1.8.2-4
libip6tc0 version: 1.8.2-4
libiptc0 version: 1.8.2-4
libisc-export1100 version: 1:9.11.5.P4+dfsg-5.1+deb10u3
libjson-c3 version: 0.12.1+ds-2+deb10u1
libk5crypto3 version: 1.17-3+deb10u1
libkeyutils1 version: 1.6-6
libklibc version: 2.0.6-1
libkmod2 version: 26-1
libkrb5-3 version: 1.17-3+deb10u1
libkrb5support0 version: 1.17-3+deb10u1
libksba8 version: 1.3.5-2
libkyotocabinet16v5 version: 1.2.76-4.2+b1
libldap-2.4-2 version: 2.4.47+dfsg-3+deb10u6
libldap-common version: 2.4.47+dfsg-3+deb10u6
liblocale-gettext-perl version: 1.07-3+b4
liblognorm5 version: 2.0.5-1
libltdl7 version: 2.4.6-9
liblz4-1 version: 1.8.3-1
liblzma5 version: 5.2.4-1
liblzo2-2 version: 2.10-0.1
libmagic-mgc version: 1:5.35-4+deb10u2
libmagic1 version: 1:5.35-4+deb10u2
libmailutils5 version: 1:3.5-4
libmariadb3 version: 1:10.3.27-0+deb10u1
libmnl0 version: 1.0.4-2
libmount1 version: 2.33.1-0.1
libmpdec2 version: 2.4.2-2
libncurses6 version: 6.1+20181013-2+deb10u2

libncursesw6 version: 6.1+20181013-2+deb10u2
libnetfilter-contrack3 version: 1.0.7-1
libnettle6 version: 3.4.1-1
libnewt0.52 version: 0.52.20-8
libnfnetwork0 version: 1.0.1-3+b1
libnftnl11 version: 1.1.2-2
libnghttp2-14 version: 1.36.0-2+deb10u1
libnpt0 version: 1.6-1
libntlm0 version: 1.5-1+deb10u1
libnuma1 version: 2.0.12-1
libopenipmi0 version: 2.0.25-2.1
libopts25 version: 1:5.18.12-4
libp11-kit0 version: 0.23.15-2+deb10u1
libpam-modules version: 1.3.1-5
libpam-modules-bin version: 1.3.1-5
libpam-runtime version: 1.3.1-5
libpam0g version: 1.3.1-5
libpci3 version: 1:3.5.2-1
libpcre3 version: 2:8.39-12
libperl5.28 version: 5.28.1-6+deb10u1
libpopt0 version: 1.16-12
libprocps7 version: 2:3.3.15-2
libpsl5 version: 0.20.2-2
libpython2.7 version: 2.7.16-2+deb10u1
libpython2.7-minimal version: 2.7.16-2+deb10u1
libpython2.7-stdlib version: 2.7.16-2+deb10u1
libpython3.7 version: 3.7.3-2+deb10u3
libpython3.7-minimal version: 3.7.3-2+deb10u3
libpython3.7-stdlib version: 3.7.3-2+deb10u3
libreadline7 version: 7.0-5
librtmp1 version: 2.4+20151223.gitfa8646d.1-2
libsasl2-2 version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules-db version: 2.1.27+dfsg-1+deb10u1
libseccomp2 version: 2.3.3-4
libselinux1 version: 2.8-1+b1
libsemanage-common version: 2.8-2
libsemanage1 version: 2.8-2
libsensors-config version: 1:3.5.0-3
libsensors5 version: 1:3.5.0-3
libsepol1 version: 2.8-1
libsgutils2-2 version: 1.44-1
libslang2 version: 2.3.2-2
libsmartcols1 version: 2.33.1-0.1
libsnmp-base version: 5.7.3+dfsg-5+deb10u2
libsnmp30 version: 5.7.3+dfsg-5+deb10u2
libsqlite3-0 version: 3.27.2-3+deb10u1
libss2 version: 1.44.5-1+deb10u3
libssh2-1 version: 1.8.0-2.1
libssl1.1 version: 1.1.1d-0+deb10u5
libstdc++6 version: 8.3.0-6
libsysfs2 version: 2.1.0+repack-5
libsystemd0 version: 241-7~deb10u7
libtasn1-6 version: 4.13-3

libtext-charwidth-perl version: 0.04-7.1+b1
libtext-iconv-perl version: 1.7-5+b7
libtext-wrapi18n-perl version: 0.06-7.1
libtinfo6 version: 6.1+20181013-2+deb10u2
libuchardet0 version: 0.0.6-3
libudev1 version: 241-7~deb10u7
libunistring2 version: 0.9.10-1
liburcu6 version: 0.10.2-1
libuuid1 version: 2.33.1-0.1
libwrap0 version: 7.6.q-28
libx86emu2 version: 2.0-1
libxml2 version: 2.9.4+dfsg1-7+deb10u1
libxtables12 version: 1.8.2-4
libyajl2 version: 2.1.0-3
libzstd1 version: 1.3.8+dfsg-3+deb10u2
linux-base version: 4.6
linux-firmware version: 1.195
login version: 1:4.5-1.1
logrotate version: 3.14.0-4
lsb-base version: 10.2019051400
lsscsi version: 0.30-0.1
mailutils version: 1:3.5-4
mailutils-common version: 1:3.5-4
mariadb-common version: 1:10.3.27-0+deb10u1
mawk version: 1.3.3-17+b3
megacli version: 8.07.14-2+Debian.buster.10
mime-support version: 3.62
mount version: 2.33.1-0.1
multipath-tools version: 0.7.9-3+deb10u1
multipath-tools-boot version: 0.7.9-3+deb10u1
mysql-common version: 5.8+1.0.5
nano version: 3.2-3
ncurses-base version: 6.1+20181013-2+deb10u2
ncurses-bin version: 6.1+20181013-2+deb10u2
net-tools version: 1.60+git20180626.aebd88e-1
netbase version: 5.6
ntp version: 1:4.2.8p12+dfsg-4
openipmi version: 2.0.25-2.1
openresolv version: 3.8.0-1
openssl version: 1.1.1d-0+deb10u5
passwd version: 1:4.5-1.1
perl version: 5.28.1-6+deb10u1
perl-base version: 5.28.1-6+deb10u1
perl-modules-5.28 version: 5.28.1-6+deb10u1
pigz version: 2.4-1
pinentry-curses version: 1.1.0-2
powermgmt-base version: 1.34
procps version: 2:3.3.15-2
publicsuffix version: 20190415.1030-1
python3.7 version: 3.7.3-2+deb10u3
python3.7-minimal version: 3.7.3-2+deb10u3
qemu-guest-agent version: 1:3.1+dfsg-8+deb10u8
readline-common version: 7.0-5
rsyslog version: 8.1901.0-1

runit-helper version: 2.8.6
sdparm version: 1.10-1
sed version: 4.7-1
sensible-utils version: 0.0.12
sg3-utils version: 1.44-1
sg3-utils-udev version: 1.44-1
shared-mime-info version: 1.10-1
smartmontools version: 6.6-1
smp-utils version: 0.98-2
snmp version: 5.7.3+dfsg-5+deb10u2
snmpd version: 5.7.3+dfsg-5+deb10u2
snmp version: 1:4.2.8p12+dfsg-4
ssmtp version: 2.64-8.1
sysstat version: 12.0.3-2
systemd version: 241-7~deb10u7
systemd-sysv version: 241-7~deb10u7
sysvinit-utils version: 2.93-8
tar version: 1.30+dfsg-6
tasksel version: 3.53
tasksel-data version: 3.53
tofrodos version: 1.7.13+ds-4
traceroute version: 1:2.1.0-2
tzdata version: 2021a-0+deb10u1
ucf version: 3.0038+nmu1
udev version: 241-7~deb10u7
util-linux version: 2.33.1-0.1
vim-common version: 2:8.1.0875-5
vim-tiny version: 2:8.1.0875-5
whiptail version: 0.52.20-8
xdg-user-dirs version: 0.17-2
xxd version: 2:8.1.0875-5
xz-utils version: 5.2.4-1
zlib1g version: 1:1.2.11.dfsg-1
python package zope.interface version: 5.4.0
python package ipaddress version: 1.0.23
python package cryptography version: 3.4.7
python package pyOpenSSL version: 20.0.1
python package service_identity version: 18.1.0
python package requests version: 2.25.1
python package incremental version: 21.3.0
python package Twisted[tls] version: 21.2.0
python package pyutil version: 3.3.0
python package python-dateutil version: 2.8.1
python package Werkzeug version: 1.0.1
python package klein version: 20.6.0
python package zfec version: 1.5.5
python package yajl-py version: 2.1.2
python package certifi
python package pyratemp version: 0.3.2
python package numpy version: 1.20.2

Swarm Storage 12.1 Release

- [New Features](#)
- [Additional Changes](#)
- [Upgrade Impacts](#)

New Features

- **Performance Gains**
 - The maximum CPU core count is increased from 64 to 512. (SWAR-9061)
 - Population of the overlay index at startup is improved which allows it to reach an authoritative state more quickly. (SWAR-9042)
 - The number of simultaneous requests to port 90 (legacy console) and port 91 (management API) are now metered. This limits the operational impact of something like an errant monitoring system. (SWAR-8985)
- **Health Processing and Monitoring**
 - Swarm now gathers and publishes incremental statistics for TCP and UDP inter-cluster communications. These statistics can be used to help pinpoint cluster-specific network issues. (SWAR-9047)
 - Swarm now performs defragmentation (trapped space reduction) more evenly throughout the health processor cycle to reduce trapped space fluctuations. (SWAR-8892)
 - A cluster can now be configured for faster volume defragmentation in cases where a backlog of disk fragmentation arises from small object turnover. Contact DataCore Support to get it configured if the cluster benefits from such a change. (SWAR-6472)
 - Swarm log messages are now tagged with unique reporting codes that facilitate troubleshooting when working with DataCore Support. (SWAR-7761)
- **Settings Updates**
 - The setting `scsp.enableVolumeRedirects` is now a persisted cluster setting. (SWAR-9003)
 - The setting `recovery.autoSuspendMissingHintedVolumes` is added to allow Support to automatically suppress false FVRs for unknown volumes that may be impacting client performance. (SWAR-9067)
- **Network Interface Details in Diagnostics Menu** – The Diagnostics Menu in the system menu has additional functionality for viewing the mapping of NIC names to real MAC addresses. The new option is under #6: Network Interface Details. (SWAR-9033)
- **Preserve settings during an upgrade** – The `configure_elasticsearch_with_swarmsearch` script now preserves several settings such as `path.data` and `network.host` from before the upgrade. (SWAR-9034)
- **Trigger maintenance mode for system console shutdown/reboot** – Restarting a Swarm node from the system menu (hardware console) now triggers maintenance mode and unifies the reboot behavior across the system menu, UI, and SNMP. (SWAR-8393)

Additional Changes

These items are other changes, including those that come from testing and user feedback.

- **OSS Versions** – See [Third-Party Components for Storage 12.1](#) for the complete listing of packages and versions for this release.
 - Linux kernel is updated to 5.4.109 (SWAR-8771)
 - Kernel firmware drivers are updated to 2021-03-03 (SWAR-8771)
 - Intel ixgbe network kernel driver is updated to 5.11.3 (SWAR-8771)
 - Prometheus Node Exporter updated to 1.1.2 (SWAR-9130)
 - Debian operating system updates included (SWAR-9027)
- **Fixed in 12.1**

- **Retiring a volume** – The retire of a volume can become stalled by a feed in a paused state. (SWAR-9096)
- **Elasticsearch record cleanup** – Elasticsearch records for named streams sometimes persist even though they are deleted in Swarm using recursive delete of the containing bucket or domain. (SWAR-9095)
- **Delay in `configure_elasticsearch_with_swarm_search.py` script** – Lack of internet access results in a delay in the `configure_elasticsearch_with_swarm_search.py` script. This is addressed; it now installs the Prometheus plugin to allow monitoring via a Grafana dashboard if internet access is available, and skips the step if internet access is not available. (SWAR-9078)
- **Issue with creating untenanted objects** – An issue in Swarm 9.0 – 12.0 prevented the creation of untenanted objects when the DNS hostname of the Gateway matched a storage domain, even if the request explicitly said not to use a domain. An empty domain query argument always forces untenanted operations. (SWAR-9074)
- **Progress stalled after a prolonged outage** – A feed (search, replication, S3 backup) stops making progress after a prolonged outage and a node reboot is required to resume progress. This is resolved in 12.1. (SWAR-9062)
- **Internal 404 Not Found and other errors** – The SCSP error counter had erroneously been including internal errors unrelated to client activity. The SCSP error stat in SNMP, metrics, and the management API now includes client requests. (SWAR-9043)
- **Eliminated unnecessary feed refreshes** – Editing a search feed via Storage UI no longer triggers an unnecessary refresh of the feed. (UIS-1073, SWAR-9024)
- **URL encoding of special characters** – Characters like "<" and ">" in Swarm redirects and location headers are now properly URL-encoded. (SWAR-9023)
- **Hanging feed SEND requests** – A feed SEND request, such as those used by Remote Synchronous Write (RSW), can hang indefinitely instead of returning an error if the feed changed to a blocked state during the request. (SWAR-9019)
- **Simultaneous domain and bucket creation via POST** – Simultaneous domain and bucket creation via POST is now prevented. Only one of these requests responds with a 201 Created response. The other requests get either a 409 Conflict or 503 Service Unavailable response. SWAR-3421)
- **Improved behavior for blocked feeds watch item** – A clearer error message draws attention to the issue for S3 backup and replication feeds that are blocked due to invalid X.509 ("SSL") certificates. (SWAR-8996)
- **Error when attempting to edit search feeds** – An error popup within the UI mentioning "respondsToLists" can appear when editing and saving a search feed. (SWAR-9065)
- **Invalid or expired Swarm licenses** – Swarm does not boot if the configured license was invalid or expired. (SWAR-9050)

Upgrade Impacts



Required

Complete the migration to Swarm 11.3 and ES 6.8.6 before upgrading to Swarm 12 if on older Elasticsearch (5.6.12 or 2.3.3). See [How to Upgrade Swarm, Upgrading from Unsupported Elasticsearch](#).

These items are changes to the product function that may require operational or development changes for integrated applications. Address the upgrade impacts for each of the versions since the one currently upgrading from:

Impacts for 12.1

- **Change in the `node.cfg` file** – The previously deprecated `sysctl` section of the `node.cfg` file is removed. Use `kernel.sysctlFileUrl` (introduced in Swarm 12.0) instead if it is necessary to set kernel runtime parameters. (SWAR-8968)

Settings changes

- Updated:
 - All `sysctl.*` settings are removed. (SWAR-8968)

- `support.reportPeriod` default is changed to 21600 (6 hours). (SWAR-8424)
- **Swarm storage node metrics are deprecated** and are replaced in the next major release by the graphs and reporting from [Grafana and Prometheus Node Exporter](#). The storage administration UI is updated to allow for metrics to be turned off.
- **Differences in `scsp.forceLegacyNonce` configuration** depending on the version upgrading from (SWAR-9020):
- **Currently running a Swarm Storage version prior to 11.1**, and upgrading to 11.1, 11.2, 11.3, 12.0 or 12.1:

Before upgrading, set `scsp.forceLegacyNonce=true` in the `node.cfg` file. After the upgrade, when the cluster is fully up, update `scsp.forceLegacyNonce=false` using `swarmctl` and change `scsp.forceLegacyNonce=false` in the `node.cfg` file.

Currently running a Swarm Storage version 11.1, 11.2, 11.3, 12.0 or 12.1 and upgrading to another version from that list:

Before upgrading, verify `scsp.forceLegacyNonce=false` is in the `node.cfg` file and verify using `swarmctl` that `scsp.forceLegacyNonce=false` in the cluster.

Use `swarmctl` to check or change settings

Use `'swarmctl -C scsp.forceLegacyNonce'` to check the value of `scsp.forceLegacyNonce`.

Use `'swarmctl -C scsp.forceLegacyNonce -V False'` to set the value to false.

For more details, see <https://support.cloud.caringo.com/tools/Tech-Support-Scripts-Bundle-swarmctl.pdf>.

Impacts for 12.0

- **Upgrading Elasticsearch** – Once on Elasticsearch 6.8.6 and using the new index as primary (see [Migrating from Older Elasticsearch](#)), proceed with your Swarm 12 upgrade to Elasticsearch 7. *Reminder:* Always upgrade Swarm Search and Metrics at the same time ES is upgraded.
- **Rolling upgrade** – During a rolling upgrade from a version older than 11.1, the mixed state in Swarm versions among nodes may cause errors in the Swarm UI, `swarmctl` tool, and management API calls. Use the legacy Admin Console (port 90) to monitor the rolling upgrade. (SWAR-8716)
- **Settings changes**
 - New: `scsp.enableVolumeRedirects` (for use with Content Gateway)
 - New: `search.numberOfShards`
 - New: `snmp.enabled`
 - Changed: `network.dnsDomain` is no longer required when `network.dnsServers` is defined; name servers may be defined without a domain. (SWAR-3415)
- **Replicated clusters** – If you use replication feeds between remote clusters, upgrade and downgrade versions of Storage in those clusters at the same time. This guarantees any objects Swarm 12 converts from replication to erasure-coding protection using version 12.0+ mechanisms are handled properly. (SWAR-8957)
- **Encryption-at-rest** – If you are about to upgrade from Swarm 11.0 or earlier and you use encryption-at-rest, contact DataCore Support to verify smoothly rolling back to the prior version if needed. (SWAR-8941)
- **Named NICs** – You need to change the `"castor_net"` kernel argument if you have defined a custom list of included NIC names. Example: `"castor_net=active-backup:eth0,eth1"` (SWAR-8021)
- **Upgrading with CSN NetBoot protection** – The streamlining of network interface handling in 12.0 can affect the upgrading of some CSN implementations. If you run NetBoot protection on a single-network CSN, all MAC addresses for the storage nodes must be included in the DHCP allow-list; if not, the Swarm 11 nodes can fail to get a DHCP network address from the CSN when upgrading to 12. Follow this one-time process if this occurs:
 1. Temporarily disable the network protection.
 2. Reboot the nodes (which assigns new IPs where needed, as available in your range).

3. Add the new MAC addresses (which you can list from the [System Menu](#)) to the DHCP allow-list, and restart the DHCP service.
 4. Re-enable network protection, and boot any storage nodes that failed to restart.
- **Invalid licenses** – Swarm 12.0 no longer supports Dell OEM-style licenses, and it does not boot if the configured license is invalid or expired. Contact DataCore Support for a new license. (SWAR-9036, SWAR-9050)
 - **Chassis ID limitation** – Before upgrading storage nodes to 12.0.x, contact DataCore Support to verify the nodes are able to join the network correctly. There is an issue with some chassis IDs preventing them from completing the boot up. This is corrected in Swarm 12.1.0. (SWAR-9121)
 - **Invalid or expired Swarm licenses** – Swarm 12.0 does not boot if the configured license is invalid or expired. Use a valid license. This is corrected in Swarm 12.1.0. (SWAR-9050)
 - **Differences in `scsp.forceLegacyNonce` configuration** depending on the version being upgraded from (SWAR-9020):
 - **If currently running a Swarm Storage version prior to 11.1**, and upgrading to 11.1, 11.2, 11.3, 12.0 or 12.1:

Before upgrading, set `scsp.forceLegacyNonce=true` in the `node.cfg` file. After the upgrade, when the cluster is fully up, update `scsp.forceLegacyNonce=false` using `swarmctl` and change `scsp.forceLegacyNonce=false` in the `node.cfg` file.

If currently running a Swarm Storage version 11.1, 11.2, 11.3, 12.0 or 12.1 and upgrading to another version from that list:

Before upgrading, verify `scsp.forceLegacyNonce=false` is in the `node.cfg` file and verify using `swarmctl` that `scsp.forceLegacyNonce=false` in the cluster.

Use `swarmctl` to check or change settings

Use `'swarmctl -C scsp.forceLegacyNonce'` to check the value of `scsp.forceLegacyNonce`.

Use `'swarmctl -C scsp.forceLegacyNonce -V False'` to set the value to `false`.

For more details, see <https://support.cloud.caringo.com/tools/Tech-Support-Scripts-Bundle-swarmctl.pdf>.

Cumulative impacts

Address all upgrade impacts for *each version released* since the version upgrading from.

Review the comprehensive *Upgrade Impacts* listed for the [Swarm Storage 11.3 Release](#).

Watch Items and Known Issues

The following watch items are known:

- A node fails to mount all disks in the node if a node mounts an encrypted volume that is missing the encryption key in the configuration. (SWAR-8762)
- S3 Backup feeds do not backup logical objects greater than 5 GB; those writes fail with a **CRITICAL** log message. (SWAR-8554)
- The chassis shuts down but does not come back up when restarting a cluster of virtual machines that are UEFI-booted (versus legacy BIOS). (SWAR-8054)
- With multipath-enabled hardware, the Swarm console Disk Volume Menu may erroneously show too many disks, having multiplied the actual disks in use by the number of possible paths to them. (SWAR-7248)

These are standing operational limitations:

- The Storage UI shows no NFS config if the Elasticsearch cluster is wiped. Contact DataCore Support for help repopulating the SwarmFS config information. (SWAR-8007)

- Any incomplete multipart uploads into a bucket leaves the parts (unnamed streams) in the domain if a bucket is deleted. To find and delete them, use the `s3cmd` utility (search the Support site for "`s3cmd`" for guidance). (SWAR-7690)
- Removing subcluster assignments in the CSN UI creates invalid config parameters that prevent the unassigned nodes from booting. (SWAR-7675)
- You may see false 404 Not Found and other SCSP errors during rolling reboot in versions 11.1 through 12.0.1. To mitigate this problem, set `scsp.forceLegacyNonce=False` in the cluster configuration. Remove this setting before upgrading to 12.1.0 or later. (SWAR-9020)
- During a node reboot, such as a rolling reboot of the cluster, a newly booted node can temporarily return an empty result set for a listing query. (SWAR-9083)
- S3 Backup restoration to the cluster may be blocked if the certificate is not located where Swarm expects it when using certificates with HAProxy. From 12.1, a clearer error message draws attention to the issue for S3 backup and replication feeds that are blocked due to invalid X.509 ("SSL") certificates. (SWAR-8996)

To upgrade Swarm 9 or higher, proceed now to [How to Upgrade Swarm](#). Contact DataCore Support for guidance if migrating from Swarm 8.x or earlier.

Third-Party Components for Storage 12.1

For licensing information, see [Open Source Software Licenses](#).

Elasticsearch and Swarm distributions

Elasticsearch 7.5.2
 Swarm S3 Backup Restore 1.2.4
 Swarm Search 7.0.1
 Swarm Metrics 7.0.1

Swarm Storage components

Operating system: Debian GNU/Linux 10 (buster)
 Linux kernel: 5.4.109
 kernel module 3w_9xxx 3ware 9000 Storage Controller Linux Driver: 2.26.02.014
 kernel module 3w_sas LSI 3ware SAS/SATA-RAID Linux Driver: 3.26.02.000
 kernel module 3w_xxxx 3ware Storage Controller Linux Driver: 1.26.02.003
 kernel module 8021q : 1.8
 kernel module 8139cp RealTek RTL-8139C+ series 10/100 PCI Ethernet driver: 1.3
 kernel module 8139too RealTek RTL-8139 Fast Ethernet driver: 0.9.28
 kernel module aacraid Dell PERC2, 2/Si, 3/Si, 3/Di, Adaptec Advanced Raid Products, HP NetRAID-4M, IBM ServeRAID & ICP SCSI driver: 1.2.1
 [50877]-custom
 kernel module acard_ahci ACard AHCI SATA low-level driver: 1.0
 kernel module ad7418 AD7416/17/18 driver: 0.4
 kernel module ahci AHCI SATA low-level driver: 3.0
 kernel module aic79xx Adaptec AIC790X U320 SCSI Host Bus Adapter driver: 3.0
 kernel module aic7xxx Adaptec AIC77XX/78XX SCSI Host Bus Adapter driver: 7.0
 kernel module aic94xx Adaptec aic94xx SAS/SATA driver: 1.0.3
 kernel module am53c974 AM53C974 SCSI driver: 1.00
 kernel module amd_xgbe AMD 10 Gigabit Ethernet Driver: 1.0.3
 kernel module arcmsr Areca ARC11xx/12xx/16xx/188x SAS/SATA RAID Controller Driver: v1.40.00.10-20190116
 kernel module ata_generic low-level driver for generic ATA: 0.2.15
 kernel module ata_piix SCSI low-level driver for Intel PIIX/ICH ATA controllers: 2.13
 kernel module atl1 Atheros L1 Gigabit Ethernet Driver: 2.1.3
 kernel module atl1c Qualcomm Atheros 100/1000M Ethernet Network Driver: 1.0.1.1-NAPI
 kernel module atl1e Atheros 1000M Ethernet Network Driver: 1.0.0.7-NAPI
 kernel module atl2 Atheros Fast Ethernet Network Driver: 2.2.3
 kernel module atlantic aQuantia Corporation(R) Network Driver: 5.4.109-kern
 kernel module atxp1 System voltages control via Attansic ATXP1: 0.6.3
 kernel module b44 Broadcom 44xx/47xx 10/100 PCI ethernet driver: 2.0
 kernel module be2iscsi Emulex OneConnectOpen-iSCSI Driver version11.4.0.1 Driver 11.4.0.1: 11.4.0.1
 kernel module be2net Emulex OneConnect NIC Driver 12.0.0.0: 12.0.0.0
 kernel module bfa QLogic BR-series Fibre Channel HBA Driver fcpim: 3.2.25.1
 kernel module bna QLogic BR-series 10G PCIe Ethernet driver: 3.2.25.1
 kernel module bnx2 QLogic BCM5706/5708/5709/5716 Driver: 2.2.6
 kernel module bnx2fc QLogic FCoE Driver: 2.12.10
 kernel module bnx2i QLogic NetXtreme II BCM5706/5708/5709/57710/57711/57712/57800/57810/57840 iSCSI Driver: 2.7.10.1
 kernel module bnx2x QLogic BCM57710/57711/57711E/57712/57712_MF/57800/57800_MF/57810/57810_MF/57840/57840_MF Driver: 1.713.36-0
 kernel module bnxt_en Broadcom BCM573xx network driver: 1.10.0

kernel module bonding Ethernet Channel Bonding Driver, v3.7.1: 3.7.1
kernel module cnic QLogic cnic Driver: 2.5.22
kernel module csiostor Chelsio FCoE driver: 1.0.0-ko
kernel module cxgb3 Chelsio T3 Network Driver: 1.1.5-ko
kernel module cxgb3i Chelsio T3 iSCSI Driver: 2.0.1-ko
kernel module cxgb4 Chelsio T4/T5/T6 Network Driver: 2.0.0-ko
kernel module cxgb4i Chelsio T4-T6 iSCSI Driver: 0.9.5-ko
kernel module cxgb4vf Chelsio T4/T5/T6 Virtual Function (VF) Network Driver: 2.0.0-ko
kernel module dca : 1.12.1
kernel module dcdbas Dell Systems Management Base Driver (version 5.6.0-3.3): 5.6.0-3.3
kernel module de2104x Intel/Digital 21040/1 series PCI Ethernet driver: 0.7
kernel module dmfe Davicom DM910X fast ethernet driver: 1.36.4
kernel module e100 Intel(R) PRO/100 Network Driver: 3.5.24-k2-NAPI
kernel module e1000 Intel(R) PRO/1000 Network Driver: 7.3.21-k8-NAPI
kernel module e1000e Intel(R) PRO/1000 Network Driver: 3.2.6-k
kernel module eeprom_93cx6 EEPROM 93cx6 chip driver: 1.0
kernel module efivars sysfs interface to EFI Variables: 0.08
kernel module ena Elastic Network Adapter (ENA): 2.1.0K
kernel module enic Cisco VIC Ethernet NIC Driver: 2.3.0.53
kernel module esas2r esas2r: 1.00
kernel module esp_scsi ESP SCSI driver core: 2.000
kernel module fm10k Intel(R) Ethernet Switch Host Interface Driver: 0.26.1-k
kernel module fnic Cisco FCoE HBA Driver: 1.6.0.47
kernel module hpsa Driver for HP Smart Array Controller version 3.4.20-170: 3.4.20-170
kernel module i40e Intel(R) 40-10 Gigabit Ethernet Connection Network Driver: 2.10.19.82
kernel module i40e Intel(R) Ethernet Connection XL710 Network Driver: 2.8.20-k
kernel module iavf Intel(R) Ethernet Adaptive Virtual Function Network Driver: 3.2.3-k
kernel module ice Intel(R) Ethernet Connection E800 Series Linux Driver: 0.8.1-k
kernel module igb Intel(R) Gigabit Ethernet Network Driver: 5.6.0-k
kernel module igbvf Intel(R) Gigabit Virtual Function Network Driver: 2.4.0-k
kernel module ioatdma : 5.00
kernel module ipmi_msghandler Incoming and outgoing message routing for an IPMI interface.: 39.2
kernel module ipr IBM Power RAID SCSI Adapter Driver: 2.6.4
kernel module ips IBM ServerAID Adapter Driver 7.12.05: 7.12.05
kernel module iscsi : 1.2.0
kernel module ixgb Intel(R) PRO/10GbE Network Driver: 1.0.135-k2-NAPI
kernel module ixgbe Intel(R) 10 Gigabit PCI Express Network Driver: 5.1.0-k
kernel module ixgbe Intel(R) 10GbE PCI Express Linux Network Driver: 5.11.3
kernel module ixgbevfn Intel(R) 10 Gigabit Virtual Function Network Driver: 4.1.0-k
kernel module jme JMicron JMC2x0 PCI Express Ethernet driver: 1.0.8
kernel module libcxgb Chelsio common library: 1.0.0-ko
kernel module libcxgbi Chelsio iSCSI driver library: 0.9.1-ko
kernel module liquidio Cavium LiquidIO Intelligent Server Adapter Driver: 1.7.2
kernel module liquidio_vf Cavium LiquidIO Intelligent Server Adapter Virtual Function Driver: 1.7.2
kernel module lpfc Emulex LightPulse Fibre Channel SCSI driver 12.4.0.0: 0
kernel module megaraid LSI Logic MegaRAID legacy driver: 2.00.4
kernel module megaraid_mbox LSI Logic MegaRAID Mailbox Driver: 2.20.5.1
kernel module megaraid_mm LSI Logic Management Module: 2.20.2.7
kernel module megaraid_sas Broadcom MegaRAID SAS Driver: 07.710.50.00-rc1
kernel module mlx4_core Mellanox ConnectX HCA low-level driver: 4.0-0
kernel module mlx4_en Mellanox ConnectX HCA Ethernet driver: 4.0-0
kernel module mlx5_core Mellanox 5th generation network adapters (ConnectX series) core driver: 5.0-0
kernel module mpt3sas LSI MPT Fusion SAS 3.0 Device Driver: 31.100.00.00
kernel module mptbase Fusion MPT base driver: 3.04.20

kernel module mptctl Fusion MPT misc device (ioctl) driver: 3.04.20
 kernel module mptfc Fusion MPT FC Host driver: 3.04.20
 kernel module mptsas Fusion MPT SAS Host driver: 3.04.20
 kernel module mptscsih Fusion MPT SCSI Host driver: 3.04.20
 kernel module mptspi Fusion MPT SPI Host driver: 3.04.20
 kernel module mtip32xx Micron RealSSD PCIe Block Driver: 1.3.1
 kernel module mvsas Marvell 88SE6440 SAS/SATA controller driver: 0.8.16
 kernel module myri10ge Myricom 10G driver (10GbE): 1.5.3-1.534
 kernel module netxen_nic QLogic/NetXen (1/10) GbE Intelligent Ethernet Driver: 4.0.82
 kernel module nfp The Netronome Flow Processor (NFP) driver.: 5.4.109
 kernel module nicpf Cavium Thunder NIC Physical Function Driver: 1.0
 kernel module nicvf Cavium Thunder NIC Virtual Function Driver: 1.0
 kernel module niu NIU ethernet driver: 1.1
 kernel module nvme : 1.0
 kernel module nvme_core : 1.0
 kernel module pata_acpi SCSI low-level driver for ATA in ACPI mode: 0.2.3
 kernel module pata_ali low-level driver for ALi PATA: 0.7.8
 kernel module pata_amd low-level driver for AMD and Nvidia PATA IDE: 0.4.1
 kernel module pata_artop SCSI low-level driver for ARTOP PATA: 0.4.6
 kernel module pata_atiixp low-level driver for ATI IXP200/300/400: 0.4.6
 kernel module pata_atp867x low level driver for Artop/Acard 867x ATA controller: 0.7.5
 kernel module pata_cmd64x low-level driver for CMD64x series PATA controllers: 0.2.18
 kernel module pata_efar SCSI low-level driver for EFAR PIIX clones: 0.4.5
 kernel module pata_hpt366 low-level driver for the Highpoint HPT366/368: 0.6.11
 kernel module pata_hpt37x low-level driver for the Highpoint HPT37x/30x: 0.6.23
 kernel module pata_hpt3x2n low-level driver for the Highpoint HPT3xxN: 0.3.15
 kernel module pata_hpt3x3 low-level driver for the Highpoint HPT343/363: 0.6.1
 kernel module pata_it821x low-level driver for the IT8211/IT8212 IDE RAID controller: 0.4.2
 kernel module pata_jmicron SCSI low-level driver for Jmicron PATA ports: 0.1.5
 kernel module pata_marvell SCSI low-level driver for Marvell ATA in legacy mode: 0.1.6
 kernel module pata_mpiix low-level driver for Intel MPIIX: 0.7.7
 kernel module pata_netcell SCSI low-level driver for Netcell PATA RAID: 0.1.7
 kernel module pata_ninja32 low-level driver for Ninja32 ATA: 0.1.5
 kernel module pata_ns87410 low-level driver for Nat Semi 87410: 0.4.6
 kernel module pata_ns87415 ATA low-level driver for NS87415 controllers: 0.0.1
 kernel module pata_oldpiix SCSI low-level driver for early PIIX series controllers: 0.5.5
 kernel module pata_pdc2027x libata driver module for Promise PDC20268 to PDC20277: 1.0
 kernel module pata_pdc202xx_old low-level driver for Promise 2024x and 20262-20267: 0.4.3
 kernel module pata_platform low-level driver for platform device ATA: 1.2
 kernel module pata_rdc SCSI low-level driver for RDC PATA controllers: 0.01
 kernel module pata_rz1000 low-level driver for RZ1000 PCI ATA: 0.2.4
 kernel module pata_sch SCSI low-level driver for Intel SCH PATA controllers: 0.2
 kernel module pata_serverworks low-level driver for Serverworks OSB4/CSB5/CSB6: 0.4.3
 kernel module pata_sil680 low-level driver for SI680 PATA: 0.4.9
 kernel module pata_sis SCSI low-level driver for SiS ATA: 0.5.2
 kernel module pata_sl82c105 low-level driver for SI82c105: 0.3.3
 kernel module pata_triflex low-level driver for Compaq Triflex: 0.2.8
 kernel module pata_via low-level driver for VIA PATA: 0.3.4
 kernel module pdc_adma Pacific Digital Corporation ADMA low-level driver: 1.0
 kernel module pm80xx PMC-Sierra PM8001/8006/8081/8088/8089/8074/8076/8077/8070/8072 SAS/SATA controller driver: 0.1.39
 kernel module pmcraid PMC Sierra MaxRAID Controller Driver: 1.0.3
 kernel module qed QLogic FastLinQ 4xxxx Core Module: 8.37.0.20
 kernel module qede QLogic FastLinQ 4xxxx Ethernet Driver: 8.37.0.20
 kernel module qedf QLogic FastLinQ 4xxxx FCoE Module: 8.42.3.0

kernel module qedi QLogic FastLinQ 4xxx iSCSI Module: 8.37.0.20
kernel module qla1280 Qlogic ISP SCSI (qla1x80/qla1x160) driver: 3.27.1
kernel module qla2xxx QLogic Fibre Channel HBA Driver: 10.01.00.19-k
kernel module qla3xxx QLogic ISP3XXX Network Driver v2.03.00-k5 : v2.03.00-k5
kernel module qla4xxx QLogic iSCSI HBA Driver: 5.04.00-k6
kernel module qlcnic QLogic 1/10 GbE Converged/Intelligent Ethernet Driver: 5.3.66
kernel module r6040 RDC R6040 NAPI PCI FastEthernet driver: 0.29 04Jul2016
kernel module rsxx IBM Flash Adapter 900GB Full Height Device Driver: 4.0.3.2516
kernel module s2io : 2.0.26.28
kernel module sata_dwc_460ex DesignWare Cores SATA controller low level driver: 1.3
kernel module sata_mv SCSI low-level driver for Marvell SATA controllers: 1.28
kernel module sata_nv low-level driver for NVIDIA nForce SATA controller: 3.5
kernel module sata_promise Promise ATA TX2/TX4/TX4000 low-level driver: 2.12
kernel module sata_qstor Pacific Digital Corporation QStor SATA low-level driver: 0.09
kernel module sata_sil low-level driver for Silicon Image SATA controller: 2.4
kernel module sata_sis low-level driver for Silicon Integrated Systems SATA controller: 1.0
kernel module sata_svw low-level driver for K2 SATA controller: 2.3
kernel module sata_sx4 Promise SATA low-level driver: 0.12
kernel module sata_uli low-level driver for ULi Electronics SATA controller: 1.3
kernel module sata_via SCSI low-level driver for VIA SATA controllers: 2.6
kernel module sata_vsc low-level driver for Vitesse VSC7174 SATA controller: 2.3
kernel module sfc Solarflare network driver: 4.1
kernel module sfc_falcon Solarflare Falcon network driver: 4.1
kernel module sg SCSI generic (sg) driver: 3.5.36
kernel module sis190 SiS sis190/191 Gigabit Ethernet driver: 1.4
kernel module skge SysKonnnect Gigabit Ethernet driver: 1.14
kernel module sky2 Marvell Yukon 2 Gigabit Ethernet driver: 1.30
kernel module slicoss Alacritech non-accelerated SLIC driver: 1.0
kernel module smartpqi Driver for Microsemi Smart Family Controller version 1.2.8-026: 1.2.8-026
kernel module smsc911x : 2008-10-21
kernel module smsc9420 : 1.01
kernel module snic Cisco SCSI NIC Driver: 0.0.1.18
kernel module starfire Adaptec Starfire Ethernet driver: 2.1
kernel module stex Promise Technology SuperTrak EX Controllers: 6.02.0000.01
kernel module sunhme Sun HappyMealEthernet(HME) 10/100baseT ethernet driver: 3.10
kernel module sym53c8xx NCR, Symbios and LSI 8xx and 1010 PCI SCSI adapters: 2.2.3
kernel module tg3 Broadcom Tigon3 ethernet driver: 3.137
kernel module thunder_bgx Cavium Thunder BGX/MAC Driver: 1.0
kernel module thunder_xcv Cavium Thunder RGX/XCV Driver: 1.0
kernel module tpm TPM Driver: 2.0
kernel module tpm_atmel TPM Driver: 2.0
kernel module tpm_crb TPM2 Driver: 0.1
kernel module tpm_i2c_infineon TPM TIS I2C Infineon Driver: 2.2.0
kernel module tpm_infineon Driver for Infineon TPM SLD 9630 TT 1.1 / SLB 9635 TT 1.2: 1.9.2
kernel module tpm_nsc TPM Driver: 2.0
kernel module tpm_st33zp24 ST33ZP24 TPM 1.2 driver: 1.3.0
kernel module tpm_st33zp24_i2c STM TPM 1.2 I2C ST33 Driver: 1.3.0
kernel module tpm_tis TPM Driver: 2.0
kernel module tpm_tis_core TPM Driver: 2.0
kernel module tpm_vtpm_proxy vTPM Driver: 0.1
kernel module tulip Digital 21*4* Tulip ethernet driver: 1.1.15
kernel module typhoon 3Com Typhoon Family (3C990, 3CR990, and variants): 1.0
kernel module ufshcd_core Generic UFS host controller driver Core: 0.2
kernel module virtio_pci virtio-pci: 1

kernel module vmw_pvscsi VMware PVSCSI driver: 1.0.7.0-k
kernel module vmxnet3 VMware vmxnet3 virtual NIC driver: 1.4.17.0-k
kernel module vxlan Driver for VXLAN encapsulated traffic: 0.1
kernel module winbond_840 Winbond W89c840 Ethernet driver: 1.01-e
adduser version: 3.118
apt version: 1.8.2.2
apt-utils version: 1.8.2.2
base-files version: 10.3+deb10u9
base-passwd version: 3.5.46
bash version: 5.0-4
bsdmainutils version: 11.1.2+b1
bsdutils version: 1:2.33.1-0.1
busybox version: 1:1.30.1-4
bzip2 version: 1.0.6-9.2~deb10u1
ca-certificates version: 20200601~deb10u2
coreutils version: 8.30-3
cpio version: 2.12+dfsg-9
cron version: 3.0pl1-134+deb10u1
cryptsetup-bin version: 2:2.1.0-5+deb10u2
curl version: 7.64.0-4+deb10u1
dash version: 0.5.10.2-5
debconf version: 1.5.71
debconf-i18n version: 1.5.71
debian-archive-keyring version: 2019.1+deb10u1
debianutils version: 4.8.6.1
dhcpcd5 version: 7.1.0-2
diffutils version: 1:3.7-3
dirmngr version: 2.2.12-1+deb10u1
dmidecode version: 3.2-1
dmsetup version: 2:1.02.155-3
dosfstools version: 4.1-2
dpkg version: 1.19.7
e2fsprogs version: 1.44.5-1+deb10u3
ethtool version: 1:4.19-1
fdisk version: 2.33.1-0.1
file version: 1:5.35-4+deb10u2
findutils version: 4.6.0+git+20190209-2
gcc-8-base version: 8.3.0-6
gdbm-l10n version: 1.18.1-4
gdisk version: 1.0.3-1.1
gnupg version: 2.2.12-1+deb10u1
gnupg-l10n version: 2.2.12-1+deb10u1
gnupg-utils version: 2.2.12-1+deb10u1
gpg version: 2.2.12-1+deb10u1
gpg-agent version: 2.2.12-1+deb10u1
gpg-wks-client version: 2.2.12-1+deb10u1
gpg-wks-server version: 2.2.12-1+deb10u1
gpgconf version: 2.2.12-1+deb10u1
gpgsm version: 2.2.12-1+deb10u1
gpgv version: 2.2.12-1+deb10u1
grep version: 3.3-1
groff-base version: 1.22.4-3+deb10u1
guile-2.2-libs version: 2.2.4+1-2+deb10u1
gzip version: 1.9-3

hdparm version: 9.58+ds-1
hostname version: 3.21
hwnfo version: 21.63-3
ifenslave version: 2.9
ifupdown version: 0.8.35
init version: 1.56+nmu1
init-system-helpers version: 1.56+nmu1
initramfs-tools version: 0.133+deb10u1
initramfs-tools-core version: 0.133+deb10u1
iproute2 version: 4.20.0-2+deb10u1
iptables version: 1.8.2-4
iputils-ping version: 3:20180629-2+deb10u2
irqbalance version: 1.5.0-3
isc-dhcp-client version: 4.4.1-2
isc-dhcp-common version: 4.4.1-2
klibc-utils version: 2.0.6-1
kmod version: 26-1
kpartx version: 0.7.9-3+deb10u1
krb5-locales version: 1.17-3+deb10u1
less version: 487-0.1+b1
libacl1 version: 2.2.53-4
libaio1 version: 0.3.112-3
libapparmor1 version: 2.13.2-10
libapt-inst2.0 version: 1.8.2.2
libapt-pkg5.0 version: 1.8.2.2
libargon2-1 version: 0~20171227-0.2
libassuan0 version: 2.5.2-1
libattr1 version: 1:2.4.48-4
libaudit-common version: 1:2.8.4-3
libaudit1 version: 1:2.8.4-3
libblkid1 version: 2.33.1-0.1
libboost-atomic1.67.0 version: 1.67.0-13+deb10u1
libboost-numpy1.67.0 version: 1.67.0-13+deb10u1
libboost-python1.67.0 version: 1.67.0-13+deb10u1
libboost-system1.67.0 version: 1.67.0-13+deb10u1
libboost-thread1.67.0 version: 1.67.0-13+deb10u1
libbsd0 version: 0.9.1-2+deb10u1
libbz2-1.0 version: 1.0.6-9.2~deb10u1
libc-bin version: 2.28-10
libc6 version: 2.28-10
libcap-ng0 version: 0.7.9-2
libcap2 version: 1:2.25-2
libcap2-bin version: 1:2.25-2
libcom-err2 version: 1.44.5-1+deb10u3
libcryptsetup12 version: 2:2.1.0-5+deb10u2
libcurl4 version: 7.64.0-4+deb10u1
libdb5.3 version: 5.3.28+dfsg1-0.5
libdebconfclient0 version: 0.249
libdevmapper1.02.1 version: 2:1.02.155-3
libdns-export1104 version: 1:9.11.5.P4+dfsg-5.1+deb10u3
libedit2 version: 3.1-20181209-1
libelf1 version: 0.176-1.1
libestr0 version: 0.1.10-2.1
libevent-core-2.1-6 version: 2.1.8-stable-4

libevent-pthreads-2.1-6 version: 2.1.8-stable-4
libexpat1 version: 2.2.6-2+deb10u1
libext2fs2 version: 1.44.5-1+deb10u3
libfastjson4 version: 0.99.8-2
libfdisk1 version: 2.33.1-0.1
libffi6 version: 3.2.1-9
libfribidi0 version: 1.0.5-3.1+deb10u1
libgc1c2 version: 1:7.6.4-0.4
libgcc1 version: 1:8.3.0-6
libgcrypt20 version: 1.8.4-5
libgdbm-compat4 version: 1.18.1-4
libgdbm6 version: 1.18.1-4
libglib2.0-0 version: 2.58.3-2+deb10u2
libglib2.0-data version: 2.58.3-2+deb10u2
libgmp10 version: 2:6.1.2+dfsg-4
libgnutls-openssl27 version: 3.6.7-4+deb10u6
libgnutls30 version: 3.6.7-4+deb10u6
libgpg-error0 version: 1.35-1
libgsasl7 version: 1.8.0-8+b2
libgssapi-krb5-2 version: 1.17-3+deb10u1
libhd21 version: 21.63-3
libhogweed4 version: 3.4.1-1
libicu63 version: 63.1-6+deb10u1
libidn11 version: 1.33-2.2
libidn2-0 version: 2.0.5-1+deb10u1
libip4tc0 version: 1.8.2-4
libip6tc0 version: 1.8.2-4
libiptc0 version: 1.8.2-4
libisc-export1100 version: 1:9.11.5.P4+dfsg-5.1+deb10u3
libjson-c3 version: 0.12.1+ds-2+deb10u1
libk5crypto3 version: 1.17-3+deb10u1
libkeyutils1 version: 1.6-6
libklibc version: 2.0.6-1
libkmod2 version: 26-1
libkrb5-3 version: 1.17-3+deb10u1
libkrb5support0 version: 1.17-3+deb10u1
libksba8 version: 1.3.5-2
libkyotocabinet16v5 version: 1.2.76-4.2+b1
libldap-2.4-2 version: 2.4.47+dfsg-3+deb10u6
libldap-common version: 2.4.47+dfsg-3+deb10u6
liblocale-gettext-perl version: 1.07-3+b4
liblognorm5 version: 2.0.5-1
libltdl7 version: 2.4.6-9
liblz4-1 version: 1.8.3-1
liblzma5 version: 5.2.4-1
liblzo2-2 version: 2.10-0.1
libmagic-mgc version: 1:5.35-4+deb10u2
libmagic1 version: 1:5.35-4+deb10u2
libmailutils5 version: 1:3.5-4
libmariadb3 version: 1:10.3.27-0+deb10u1
libmnl0 version: 1.0.4-2
libmount1 version: 2.33.1-0.1
libmpdec2 version: 2.4.2-2
libncurses6 version: 6.1+20181013-2+deb10u2

libncursesw6 version: 6.1+20181013-2+deb10u2
libnetfilter-contrack3 version: 1.0.7-1
libnettle6 version: 3.4.1-1
libnewt0.52 version: 0.52.20-8
libnfnetwork0 version: 1.0.1-3+b1
libnftnl11 version: 1.1.2-2
libnghttp2-14 version: 1.36.0-2+deb10u1
libnptl0 version: 1.6-1
libntlm0 version: 1.5-1+deb10u1
libnuma1 version: 2.0.12-1
libopenipmi0 version: 2.0.25-2.1
libopts25 version: 1:5.18.12-4
libp11-kit0 version: 0.23.15-2+deb10u1
libpam-modules version: 1.3.1-5
libpam-modules-bin version: 1.3.1-5
libpam-runtime version: 1.3.1-5
libpam0g version: 1.3.1-5
libpci3 version: 1:3.5.2-1
libpcre3 version: 2:8.39-12
libperl5.28 version: 5.28.1-6+deb10u1
libpopt0 version: 1.16-12
libprocps7 version: 2:3.3.15-2
libpsl5 version: 0.20.2-2
libpython2.7 version: 2.7.16-2+deb10u1
libpython2.7-minimal version: 2.7.16-2+deb10u1
libpython2.7-stdlib version: 2.7.16-2+deb10u1
libpython3.7 version: 3.7.3-2+deb10u3
libpython3.7-minimal version: 3.7.3-2+deb10u3
libpython3.7-stdlib version: 3.7.3-2+deb10u3
libreadline7 version: 7.0-5
librtmp1 version: 2.4+20151223.gitfa8646d.1-2
libsasl2-2 version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules-db version: 2.1.27+dfsg-1+deb10u1
libseccomp2 version: 2.3.3-4
libselinux1 version: 2.8-1+b1
libsemanage-common version: 2.8-2
libsemanage1 version: 2.8-2
libsensors-config version: 1:3.5.0-3
libsensors5 version: 1:3.5.0-3
libsepol1 version: 2.8-1
libsgutils2-2 version: 1.44-1
libslang2 version: 2.3.2-2
libsmartcols1 version: 2.33.1-0.1
libsnmp-base version: 5.7.3+dfsg-5+deb10u2
libsnmp30 version: 5.7.3+dfsg-5+deb10u2
libsqlite3-0 version: 3.27.2-3+deb10u1
libss2 version: 1.44.5-1+deb10u3
libssh2-1 version: 1.8.0-2.1
libssl1.1 version: 1.1.1d-0+deb10u5
libstdc++6 version: 8.3.0-6
libsysfs2 version: 2.1.0+repack-5
libsystemd0 version: 241-7~deb10u7
libtasn1-6 version: 4.13-3

libtext-charwidth-perl version: 0.04-7.1+b1
libtext-iconv-perl version: 1.7-5+b7
libtext-wrapi18n-perl version: 0.06-7.1
libtinfo6 version: 6.1+20181013-2+deb10u2
libuchardet0 version: 0.0.6-3
libudev1 version: 241-7~deb10u7
libunistring2 version: 0.9.10-1
liburcu6 version: 0.10.2-1
libuuid1 version: 2.33.1-0.1
libwrap0 version: 7.6.q-28
libx86emu2 version: 2.0-1
libxml2 version: 2.9.4+dfsg1-7+deb10u1
libxtables12 version: 1.8.2-4
libyajl2 version: 2.1.0-3
libzstd1 version: 1.3.8+dfsg-3+deb10u2
linux-base version: 4.6
linux-firmware version: 1.195
login version: 1:4.5-1.1
logrotate version: 3.14.0-4
lsb-base version: 10.2019051400
lsscsi version: 0.30-0.1
mailutils version: 1:3.5-4
mailutils-common version: 1:3.5-4
mariadb-common version: 1:10.3.27-0+deb10u1
mawk version: 1.3.3-17+b3
megacli version: 8.07.14-2+Debian.buster.10
mime-support version: 3.62
mount version: 2.33.1-0.1
multipath-tools version: 0.7.9-3+deb10u1
multipath-tools-boot version: 0.7.9-3+deb10u1
mysql-common version: 5.8+1.0.5
nano version: 3.2-3
ncurses-base version: 6.1+20181013-2+deb10u2
ncurses-bin version: 6.1+20181013-2+deb10u2
net-tools version: 1.60+git20180626.aebd88e-1
netbase version: 5.6
ntp version: 1:4.2.8p12+dfsg-4
openipmi version: 2.0.25-2.1
openresolv version: 3.8.0-1
openssl version: 1.1.1d-0+deb10u5
passwd version: 1:4.5-1.1
perl version: 5.28.1-6+deb10u1
perl-base version: 5.28.1-6+deb10u1
perl-modules-5.28 version: 5.28.1-6+deb10u1
pigz version: 2.4-1
pinentry-curses version: 1.1.0-2
powermgmt-base version: 1.34
procps version: 2:3.3.15-2
publicsuffix version: 20190415.1030-1
python3.7 version: 3.7.3-2+deb10u3
python3.7-minimal version: 3.7.3-2+deb10u3
qemu-guest-agent version: 1:3.1+dfsg-8+deb10u8
readline-common version: 7.0-5
rsyslog version: 8.1901.0-1

runit-helper version: 2.8.6
sdparm version: 1.10-1
sed version: 4.7-1
sensible-utils version: 0.0.12
sg3-utils version: 1.44-1
sg3-utils-udev version: 1.44-1
shared-mime-info version: 1.10-1
smartmontools version: 6.6-1
smp-utils version: 0.98-2
snmp version: 5.7.3+dfsg-5+deb10u2
snmpd version: 5.7.3+dfsg-5+deb10u2
snmp version: 1:4.2.8p12+dfsg-4
ssmtp version: 2.64-8.1
sysstat version: 12.0.3-2
systemd version: 241-7~deb10u7
systemd-sysv version: 241-7~deb10u7
sysvinit-utils version: 2.93-8
tar version: 1.30+dfsg-6
tasksel version: 3.53
tasksel-data version: 3.53
tofrodo version: 1.7.13+ds-4
traceroute version: 1:2.1.0-2
tzdata version: 2021a-0+deb10u1
ucf version: 3.0038+nmu1
udev version: 241-7~deb10u7
util-linux version: 2.33.1-0.1
vim-common version: 2:8.1.0875-5
vim-tiny version: 2:8.1.0875-5
whiptail version: 0.52.20-8
xdg-user-dirs version: 0.17-2
xxd version: 2:8.1.0875-5
xz-utils version: 5.2.4-1
zlib1g version: 1:1.2.11.dfsg-1

Swarm Storage 12.0 Release

- [New Features](#)
- [Additional Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Known Issues](#)
- [Upgrading Swarm](#)

New Features

Performance Gains – Swarm 12 offers overall performance improvements, most pronounced through Gateway, and for small object GETs.

- By enabling the new `scsp.enableVolumeRedirects` setting, you allow Gateway to perform redirects of GET requests to volume processes, for greater efficiency. If you use SCSP directly and want to change your client application to take advantage of these redirects, contact DataCore Support. (SWAR-8758) (CLOUD-3205)
- Inter-process communication within the cluster has been significantly streamlined, which boosts performance in dense clusters. (SWAR-8940)
- Connection balancing between SCSP processes is improved, which helps performance. (SWAR-8933)
- When there are surges in new SCSP connections, Swarm can now preemptively close new SCSP connections to protect the target node from crashing. (SWAR-8965)
- Performance under high loads has been improved, including including 503, 404, and file descriptor exhaustion. (SWAR-8971, SWAR-8969)

S3 Backup Improvements – Swarm 12 includes significant expansion of S3 Backup capabilities:

- **Backup to Glacier:** To lower the cost of disaster recovery, Swarm S3 Backup feeds can now target buckets that use non-standard "cold" storage classes, AWS S3 Glacier and S3 Glacier Deep Archive. Note: Glacier may be more cost-effective at scale due to rounding policies of Deep Archive. See [S3 Backup Feeds](#). (SWAR-8923)
- When the S3 Restore tool recovers data from backup buckets that use Glacier storage classes, it uses additional configuration settings to support retrieval from archives. Recovery from cold storage may need multiple runs to complete. See [S3 Backup Restore Tool](#). (SWAR-8967)
- For the AWS transition to virtual hosted-style URLs, Swarm S3 Backup now supports the bucket-in-host request style. In the S3 Backup feed definition, "host" and "bucket" are still entered separately. (SWAR-8917)
- For greater efficiency, S3 Backup feeds now skip backing up objects that the health processor has queued to delete. (SWAR-8931)
- S3 Backup feeds have better logging and error handling, and the S3 Restore tool has improved messaging. (SWAR-8905, SWAR-8962, SWAR-8960).

Elasticsearch 7 – Swarm 12.0 ships with and uses Elasticsearch 7.5.2, along with new versions of Swarm Search and Metrics RPMs. Upgrading requires no reindexing of your ES 6.8.6 data, so you can upgrade Elasticsearch in place, using the configuration script provided. (SWAR-8894, SWAR-8893).

- **Shard control** – The new Swarm setting, `search.numberOfShards`, allows adjusting the number of shards you want on new search indices as you scale your implementation (see [Scaling Elasticsearch](#)). The setting has no effect on existing indices; to change the shard count, create a new search feed or delete the existing ES index and Refresh the feed. See (SWAR-7276).

Feed Logging and Diagnostics – This release reworked logging to help you manage your feeds:

- Replication feeds now have improved diagnostic logging for Gateway and proxy errors. (SWAR-8951, SWAR-8811)
- Feeds that report "persistently failing" errors have better information to help with troubleshooting. (SWAR-8829)
- Improved logging helps identify connection problems with Elasticsearch. (SWAR-8909)
- Swarm monitors for Elasticsearch indices put in a read-only state due to insufficient file space on one or more Elasticsearch nodes. (SWAR-8944)

Networking and Booting – Multiple improvements to boot processes have made cluster starts faster, leaner, and sturdier:

- DHCP lease management is improved, resulting in faster boot times. (SWAR-8867)
- Boot times for VMs are faster because of better initialization of the kernel entropy pool. (SWAR-8926)
- NTP handling is improved in cases where `network.timeSource` is unspecified. (SWAR-8987)
- Volumes using encryption-at-rest have better handling and future-proofing for upgrades. (SWAR-8941)
- For clarity, network interface names now report as the native Linux kernel names and are no longer renamed to legacy "eth*" names. These native NIC names are referenced by the System Console menu, SNMP, and Prometheus. (SWAR-8021)
- With the new setting, `snmp.enabled`, you can now disable SNMP cluster-wide, which supports use of containers. (SWAR-8898)

Health Processing and Monitoring – Several enhancements support health processing and cluster administration:

- Defragmentation to release trapped space is stopped when a volume is too full for it to proceed effectively. This does not affect the volume's ability to offload content. (SWAR-8787)
- SwarmFS object uploads that are stalled "in progress" now timeout to allow consolidation and clean up of the uploaded parts. (SWAR-7699).
- To help anticipate problems with storage drives, the driveTable in SNMP has three new columns: drivePowerOnHours (the drive's power-on hours), driveTempC (the drive's temperature in Celsius), and driveCompromisedCount (the sum of five SMART values; a non-zero sum may indicate an impending drive failure). (SWAR-8734)
- Numerous improvements aid in support, such as crash handling, crash reporting, and clearer dmesg dumps. (SWAR-8979, SWAR-8988, SWAR-8798).
- The Support tool `swarmctl`, which you can download as part of the Swarm Support Tool bundle ([swarm-support-tools.tgz](#)), has expanded support for cluster capacity alerting, SMART dumps, and volume tests. (SWAR-8806, SWAR-8731, SWAR-8769).
- Swarm generates trimmer logs, having removed overly frequent ERROR messages. (SWAR-8840)

Additional Changes

These items are other changes, including those that come from testing and user feedback.

- **OSS Versions** – See [Third-Party Components for Storage 12.0](#) for the complete listing of packages and versions for this release.
 - The Linux kernel is upgraded to 5.4.61 and firmware is upgraded to 1.190. (SWAR-8956)
 - Intel network drivers i40e and ixgbe are updated. (SWAR-8845)
 - Debian 10 ("Buster") updates are incorporated into this version. (SWAR-8788)
- **Fixed in 12.0**
 - An issue related to memory corruption can result in spurious errors and false 404 Not Found responses in some cases. (12.0.1: SWAR-9077)
 - In Swarm 12.0, range reads requests during high loads may return results of the correct length but with an erroneous carriage return (CR) and line feed (LF) character inserted at the beginning of the body. Contact DataCore Support if you experience this issue. (12.0.1: SWAR-9045)
 - A node rebooting into a cluster with a different IP address appeared as offline under its former IP address in the Swarm UI and the legacy Admin Console. (SWAR-8955)
 - Volume retires can become stuck due to remaining objects needing lifepoint or other EC-related conversions. (SWAR-8945)
 - The legacy Admin Console now supports deletion of more than one feed at a time. (SWAR-8805)
 - Infrequent WARNING messages may appear in logs: "Node/Volume entry not published due to lock contention (...); action will be retried." (SWAR-8802)

Upgrade Impacts



Required

Complete the migration to Swarm 11.3 and ES 6.8.6 before upgrading to Swarm 12 if running Elasticsearch 5.6.12 or 2.3.3. See [How to Upgrade Swarm](#), *Upgrading from Unsupported Elasticsearch*.

These items are changes to the product function that may require operational or development changes for integrated applications. Address the upgrade impacts for each of the versions since the one you are currently running:

Impacts for 12.0

- **Upgrading Elasticsearch** – Once on Elasticsearch 6.8.6 and using the new index as primary (see [Migrating from Older Elasticsearch](#)), proceed with your Swarm 12 upgrade to Elasticsearch 7. *Reminder:* Always upgrade Swarm Search and Metrics at the same time ES is upgraded.
- **Rolling upgrade** – During a rolling upgrade from a version older than 11.1, the mixed state in Swarm versions among nodes may cause errors in the Swarm UI, `swarmctl` tool, and management API calls. Use the legacy Admin Console (port 90) to monitor the rolling upgrade. (SWAR-8716)
- **Settings changes**
 - New: `scsp.enableVolumeRedirects` (for use with Content Gateway)
 - New: `search.numberOfShards`
 - New: `snmp.enabled`
 - Changed: `network.dnsDomain` is no longer required when `network.dnsServers` is defined; name servers may be defined without a domain. (SWAR-3415)
- **Replicated clusters** – If you use replication feeds between remote clusters, upgrade and downgrade versions of Storage in those clusters at the same time. This guarantees any objects Swarm 12 converts from replication to erasure-coding protection using version 12.0+ mechanisms are handled properly. (SWAR-8957)
- **Encryption-at-rest** – If you are about to upgrade from Swarm 11.0 or earlier and you use encryption-at-rest, contact DataCore Support to verify smoothly rolling back to the prior version if needed. (SWAR-8941)
- **Named NICs** – You need to change the "castor_net" kernel argument if you have defined a custom list of included NIC names. Example: "castor_net=active-backup:eth0,eth1" (SWAR-8021)
- **Upgrading with CSN NetBoot protection** – The streamlining of network interface handling in 12.0 can affect the upgrading of some CSN implementations. If you run NetBoot protection on a single-network CSN, all MAC addresses for the storage nodes must be included in the DHCP allow-list; if not, the Swarm 11 nodes can fail to get a DHCP network address from the CSN when upgrading to 12. Follow this one-time process if this occurs:
 1. Temporarily disable the network protection.
 2. Reboot the nodes (which assigns new IPs where needed, as available in your range).
 3. Add the new MAC addresses (which you can list from the [System Menu](#)) to the DHCP allow-list, and restart the DHCP service.
 4. Re-enable network protection, and boot any storage nodes that failed to restart.
- **Invalid licenses** – Swarm 12.0 no longer supports Dell OEM-style licenses, and it does not boot if the configured license is invalid or expired. Contact DataCore Support for a new license. (SWAR-9036, SWAR-9050)
- **Chassis ID limitation** – Before upgrading storage nodes to 12.0.x, contact DataCore Support to verify the nodes are able to join the network correctly. There is an issue with some chassis IDs preventing them from completing the boot up. This is corrected in Swarm 12.1.0. (SWAR-9121)
- **Invalid or expired Swarm licenses** – Swarm 12.0 does not boot if the configured license is invalid or expired. Use a valid license. This is corrected in Swarm 12.1.0. (SWAR-9050)

- **Differences in `scsp.forceLegacyNonce` configuration** depending on the version being upgraded from (SWAR-9020):
- **If currently running a Swarm Storage version prior to 11.1**, and upgrading to 11.1, 11.2, 11.3, 12.0 or 12.1:

Before upgrading, set `scsp.forceLegacyNonce=true` in the `node.cfg` file. After the upgrade, when the cluster is fully up, update `scsp.forceLegacyNonce=false` using `swarmctl` and change `scsp.forceLegacyNonce=false` in the `node.cfg` file.

If currently running a Swarm Storage version 11.1, 11.2, 11.3, 12.0 or 12.1 and upgrading to another version from that list:

Before upgrading, verify `scsp.forceLegacyNonce=false` is in the `node.cfg` file and verify using `swarmctl` that `scsp.forceLegacyNonce=false` in the cluster.

Use `swarmctl` to check or change settings

Use `'swarmctl -C scsp.forceLegacyNonce'` to check the value of `scsp.forceLegacyNonce`.

Use `'swarmctl -C scsp.forceLegacyNonce -V False'` to set the value to false.

For more details, see <https://support.cloud.caringo.com/tools/Tech-Support-Scripts-Bundle-swarmctl.pdf>.

Cumulative impacts

Address all upgrade impacts for *each version released* since the version upgrading from.

Review the comprehensive *Upgrade Impacts* listed for the [Swarm Storage 11.3 Release](#).

Watch Items and Known Issues

The following watch items are known:

- When you use certificates with HAProxy, S3 Backup restoration to the cluster may be blocked if the certificate is not located where Swarm expects it. (SWAR-8996)
- If a node mounts an encrypted volume that is missing the encryption key in the configuration, the node fails to mount all disks in the node. (SWAR-8762)
- S3 Backup feeds do not backup logical objects greater than 5 GB; those writes fail with a CRITICAL log message. (SWAR-8554)
- When restarting a cluster of virtual machines that are UEFI-booted (versus legacy BIOS), the chassis shut down but do not come back up. (SWAR-8054)
- With multipath-enabled hardware, the Swarm console Disk Volume Menu may erroneously show too many disks, having multiplied the actual disks in use by the number of possible paths to them. (SWAR-7248)

These are standing operational limitations:

- If you wipe your Elasticsearch cluster, the Storage UI shows no NFS config. Contact DataCore Support for help repopulating your SwarmFS config information. (SWAR-8007)
- If you delete a bucket, any incomplete multipart upload into that bucket leaves the parts (unnamed streams) in the domain. To find and delete them, use the `s3cmd` utility (search the Support site for "s3cmd" for guidance). (SWAR-7690)
- Removing subcluster assignments in the CSN UI creates invalid config parameters preventing unassigned nodes from booting. (SWAR-7675)

- You may see false 404 Not Found and other SCSP errors during rolling reboot in versions 11.1 through 12.0.1. To mitigate this problem, set `scsp.forceLegacyNonce=False` in the cluster configuration. You need to remove this setting before upgrading to 12.1.0 or later. (SWAR-9020)
- A feed SEND request for a replication feed that is changing its state to "blocked" during the request can potentially run indefinitely, rather than given an error condition. This may impact the Remote Synchronous Write (RSW) feature used by the Gateway. This issue is addressed in Swarm 12.1.0. (SWAR-9019)
- During a node reboot, such as a rolling reboot of the cluster, a newly booted node can temporarily return an empty result set for a listing query. (SWAR-9083)
- If a feed is subject to a prolonged outage, a node reboot may be required for it to resume progress after the outage is cleared. If progress is not resolved after the reboot, contact DataCore Support. This has been resolved in 12.1.0 (SWAR-9062)
- When editing and saving a search feed in Swarm UI you may get a red error box mentioning "respondsToLists". You need to use the Swarm console instead to edit this feed. (SWAR-9065)

Upgrading Swarm

Consider these installation issues when upgrading Swarm:

- The `elasticsearch-curator` package may show an error during an upgrade, which is a known curator issue. Workaround: Reinstall the curator: `yum reinstall elasticsearch-curator` (SWAR-7439)
- Do not install the Swarm Search RPM before installing Java. If Gateway startup fails with "Caringo script plugin is missing from indexer nodes", uninstall and reinstall the Swarm Search RPM. (SWAR-7688)

Proceed to [How to Upgrade Swarm](#) to upgrade Swarm 9 or higher. Contact DataCore Support for guidance if needing to migrate from Swarm 8.x or earlier.

Third-Party Components for Storage 12.0

For licensing information, see [Open Source Software Licenses](#).

Elasticsearch and Caringo distributions

Elasticsearch 7.5.2
elasticsearch-curator 4.3.1 (supports Elasticsearch 5 and higher)
txes 0.1.4+
Swarm S3 Backup Restore 1.2.3
Swarm Search 7.0.0
Swarm Metrics 7.0.0

Swarm Storage components

zope.interface version: 4.7.1
ipaddress version: 1.0.23
cryptography version: 2.8
pyOpenSSL version: 19.1.0
service_identity version: 18.1.0
incremental version: 17.5.0
Twisted[tls] version: 19.10.0
pyutil version: 3.3.0
python-dateutil version: 2.8.1
Werkzeug version: 0.16.0
klein version: 19.6.0
requests version: 2.21.0
zfec version: 1.5.3
yajl-py version: 2.1.2
certifi version: *latest as of 2020-11-23*
pyratemp version: 0.3.2
Newt version: 0.52.20
Prometheus node_exporter version: 0.18.1
libpng version: 1.2.8
LILO version: 22.7.1
Operating system: Debian GNU/Linux 10 (buster)
Linux kernel: 5.4.61
kernel module 3w_9xxx 3ware 9000 Storage Controller Linux Driver: 2.26.02.014
kernel module 3w_sas LSI 3ware SAS/SATA-RAID Linux Driver: 3.26.02.000
kernel module 3w_xxxx 3ware Storage Controller Linux Driver: 1.26.02.003
kernel module 8021q : 1.8
kernel module 8139cp RealTek RTL-8139C+ series 10/100 PCI Ethernet driver: 1.3
kernel module 8139too RealTek RTL-8139 Fast Ethernet driver: 0.9.28
kernel module aacraid Dell PERC2, 2/Si, 3/Si, 3/Di, Adaptec Advanced Raid Products, HP NetRAID-4M, IBM ServeRAID & ICP SCSI driver: 1.2.1
[50877]-custom
kernel module acard_ahci ACard AHCI SATA low-level driver: 1.0
kernel module ad7418 AD7416/17/18 driver: 0.4
kernel module ahci AHCI SATA low-level driver: 3.0
kernel module aic79xx Adaptec AIC790X U320 SCSI Host Bus Adapter driver: 3.0
kernel module aic7xxx Adaptec AIC77XX/78XX SCSI Host Bus Adapter driver: 7.0
kernel module aic94xx Adaptec aic94xx SAS/SATA driver: 1.0.3

kernel module am53c974 AM53C974 SCSI driver: 1.00
kernel module amd_xgbe AMD 10 Gigabit Ethernet Driver: 1.0.3
kernel module arcmsr Areca ARC11xx/12xx/16xx/188x SAS/SATA RAID Controller Driver: v1.40.00.10-20190116
kernel module ata_generic low-level driver for generic ATA: 0.2.15
kernel module ata_piix SCSI low-level driver for Intel PIIX/ICH ATA controllers: 2.13
kernel module atl1 Atheros L1 Gigabit Ethernet Driver: 2.1.3
kernel module atl1c Qualcomm Atheros 100/1000M Ethernet Network Driver: 1.0.1.1-NAPI
kernel module atl1e Atheros 1000M Ethernet Network Driver: 1.0.0.7-NAPI
kernel module atl2 Atheros Fast Ethernet Network Driver: 2.2.3
kernel module atlantic aQuantia Corporation(R) Network Driver: 5.4.61-kern
kernel module atxp1 System voltages control via Attansic ATXP1: 0.6.3
kernel module b44 Broadcom 44xx/47xx 10/100 PCI ethernet driver: 2.0
kernel module be2iscsi Emulex OneConnectOpen-iSCSI Driver version11.4.0.1 Driver 11.4.0.1: 11.4.0.1
kernel module be2net Emulex OneConnect NIC Driver 12.0.0.0: 12.0.0.0
kernel module bfa QLogic BR-series Fibre Channel HBA Driver fcpim: 3.2.25.1
kernel module bna QLogic BR-series 10G PCIe Ethernet driver: 3.2.25.1
kernel module bnx2 QLogic BCM5706/5708/5709/5716 Driver: 2.2.6
kernel module bnx2fc QLogic FCoE Driver: 2.12.10
kernel module bnx2i QLogic NetXtreme II BCM5706/5708/5709/57710/57711/57712/57800/57810/57840 iSCSI Driver: 2.7.10.1
kernel module bnx2x QLogic BCM57710/57711/57711E/57712/57712_MF/57800/57800_MF/57810/57810_MF/57840/57840_MF Driver: 1.713.36-0
kernel module bnxt_en Broadcom BCM573xx network driver: 1.10.0
kernel module bonding Ethernet Channel Bonding Driver, v3.7.1: 3.7.1
kernel module cnic QLogic cnic Driver: 2.5.22
kernel module csiostor Chelsio FCoE driver: 1.0.0-ko
kernel module cxgb3 Chelsio T3 Network Driver: 1.1.5-ko
kernel module cxgb3i Chelsio T3 iSCSI Driver: 2.0.1-ko
kernel module cxgb4 Chelsio T4/T5/T6 Network Driver: 2.0.0-ko
kernel module cxgb4i Chelsio T4-T6 iSCSI Driver: 0.9.5-ko
kernel module cxgb4vf Chelsio T4/T5/T6 Virtual Function (VF) Network Driver: 2.0.0-ko
kernel module dca : 1.12.1
kernel module dcdbas Dell Systems Management Base Driver (version 5.6.0-3.3): 5.6.0-3.3
kernel module de2104x Intel/Digital 21040/1 series PCI Ethernet driver: 0.7
kernel module dmfe Davicom DM910X fast ethernet driver: 1.36.4
kernel module e100 Intel(R) PRO/100 Network Driver: 3.5.24-k2-NAPI
kernel module e1000 Intel(R) PRO/1000 Network Driver: 7.3.21-k8-NAPI
kernel module e1000e Intel(R) PRO/1000 Network Driver: 3.2.6-k
kernel module eeprom_93cx6 EEPROM 93cx6 chip driver: 1.0
kernel module efivars sysfs interface to EFI Variables: 0.08
kernel module ena Elastic Network Adapter (ENA): 2.1.0K
kernel module enic Cisco VIC Ethernet NIC Driver: 2.3.0.53
kernel module esas2r esas2r: 1.00
kernel module esp_scsi ESP SCSI driver core: 2.000
kernel module fm10k Intel(R) Ethernet Switch Host Interface Driver: 0.26.1-k
kernel module fnic Cisco FCoE HBA Driver: 1.6.0.47
kernel module hpsa Driver for HP Smart Array Controller version 3.4.20-170: 3.4.20-170
kernel module i40e Intel(R) 40-10 Gigabit Ethernet Connection Network Driver: 2.10.19.82
kernel module i40e Intel(R) Ethernet Connection XL710 Network Driver: 2.8.20-k
kernel module iavf Intel(R) Ethernet Adaptive Virtual Function Network Driver: 3.2.3-k
kernel module ice Intel(R) Ethernet Connection E800 Series Linux Driver: 0.8.1-k
kernel module igb Intel(R) Gigabit Ethernet Network Driver: 5.6.0-k
kernel module igbvf Intel(R) Gigabit Virtual Function Network Driver: 2.4.0-k
kernel module ioatdma : 5.00
kernel module ipmi_msghandler Incoming and outgoing message routing for an IPMI interface.: 39.2

kernel module ipr IBM Power RAID SCSI Adapter Driver: 2.6.4
kernel module ips IBM ServerRAID Adapter Driver 7.12.05: 7.12.05
kernel module iscsi : 1.2.0
kernel module ixgb Intel(R) PRO/10GbE Network Driver: 1.0.135-k2-NAPI
kernel module ixgbe Intel(R) 10 Gigabit PCI Express Network Driver: 5.1.0-k
kernel module ixgbe Intel(R) 10GbE PCI Express Linux Network Driver: 5.7.1
kernel module ixgbevf Intel(R) 10 Gigabit Virtual Function Network Driver: 4.1.0-k
kernel module jme JMicron JMC2x0 PCI Express Ethernet driver: 1.0.8
kernel module libcxgb Chelsio common library: 1.0.0-ko
kernel module libcxgbi Chelsio iSCSI driver library: 0.9.1-ko
kernel module liquidio Cavium LiquidIO Intelligent Server Adapter Driver: 1.7.2
kernel module liquidio_vf Cavium LiquidIO Intelligent Server Adapter Virtual Function Driver: 1.7.2
kernel module lpfc Emulex LightPulse Fibre Channel SCSI driver 12.4.0.0: 0
kernel module megaraid LSI Logic MegaRAID legacy driver: 2.00.4
kernel module megaraid_mbox LSI Logic MegaRAID Mailbox Driver: 2.20.5.1
kernel module megaraid_mm LSI Logic Management Module: 2.20.2.7
kernel module megaraid_sas Broadcom MegaRAID SAS Driver: 07.710.50.00-rc1
kernel module mlx4_core Mellanox ConnectX HCA low-level driver: 4.0-0
kernel module mlx4_en Mellanox ConnectX HCA Ethernet driver: 4.0-0
kernel module mlx5_core Mellanox 5th generation network adapters (ConnectX series) core driver: 5.0-0
kernel module mpt3sas LSI MPT Fusion SAS 3.0 Device Driver: 31.100.00.00
kernel module mptbase Fusion MPT base driver: 3.04.20
kernel module mptctl Fusion MPT misc device (ioctl) driver: 3.04.20
kernel module mptfc Fusion MPT FC Host driver: 3.04.20
kernel module mptsas Fusion MPT SAS Host driver: 3.04.20
kernel module mptscsih Fusion MPT SCSI Host driver: 3.04.20
kernel module mptspi Fusion MPT SPI Host driver: 3.04.20
kernel module mtip32xx Micron RealSSD PCIe Block Driver: 1.3.1
kernel module mvsas Marvell 88SE6440 SAS/SATA controller driver: 0.8.16
kernel module myri10ge Myricom 10G driver (10GbE): 1.5.3-1.534
kernel module netxen_nic QLogic/NetXen (1/10) GbE Intelligent Ethernet Driver: 4.0.82
kernel module nfp The Netronome Flow Processor (NFP) driver.: 5.4.61
kernel module nicpf Cavium Thunder NIC Physical Function Driver: 1.0
kernel module nicvf Cavium Thunder NIC Virtual Function Driver: 1.0
kernel module niu NIU ethernet driver: 1.1
kernel module nvme : 1.0
kernel module nvme_core : 1.0
kernel module pata_acpi SCSI low-level driver for ATA in ACPI mode: 0.2.3
kernel module pata_ali low-level driver for ALi PATA: 0.7.8
kernel module pata_amd low-level driver for AMD and Nvidia PATA IDE: 0.4.1
kernel module pata_artop SCSI low-level driver for ARTOP PATA: 0.4.6
kernel module pata_atiixp low-level driver for ATI IXP200/300/400: 0.4.6
kernel module pata_atp867x low level driver for Artop/Acard 867x ATA controller: 0.7.5
kernel module pata_cmd64x low-level driver for CMD64x series PATA controllers: 0.2.18
kernel module pata_efar SCSI low-level driver for EFAR PIIX clones: 0.4.5
kernel module pata_hpt366 low-level driver for the Highpoint HPT366/368: 0.6.11
kernel module pata_hpt37x low-level driver for the Highpoint HPT37x/30x: 0.6.23
kernel module pata_hpt3x2n low-level driver for the Highpoint HPT3xxN: 0.3.15
kernel module pata_hpt3x3 low-level driver for the Highpoint HPT343/363: 0.6.1
kernel module pata_it821x low-level driver for the IT8211/IT8212 IDE RAID controller: 0.4.2
kernel module pata_jmicron SCSI low-level driver for Jmicron PATA ports: 0.1.5
kernel module pata_marvell SCSI low-level driver for Marvell ATA in legacy mode: 0.1.6
kernel module pata_mpiix low-level driver for Intel MPIIX: 0.7.7
kernel module pata_netcell SCSI low-level driver for Netcell PATA RAID: 0.1.7

kernel module pata_ninja32 low-level driver for Ninja32 ATA: 0.1.5
kernel module pata_ns87410 low-level driver for Nat Semi 87410: 0.4.6
kernel module pata_ns87415 ATA low-level driver for NS87415 controllers: 0.0.1
kernel module pata_oldpiix SCSI low-level driver for early PIIX series controllers: 0.5.5
kernel module pata_pdc2027x libata driver module for Promise PDC20268 to PDC20277: 1.0
kernel module pata_pdc202xx_old low-level driver for Promise 2024x and 20262-20267: 0.4.3
kernel module pata_platform low-level driver for platform device ATA: 1.2
kernel module pata_rdc SCSI low-level driver for RDC PATA controllers: 0.01
kernel module pata_rz1000 low-level driver for RZ1000 PCI ATA: 0.2.4
kernel module pata_sch SCSI low-level driver for Intel SCH PATA controllers: 0.2
kernel module pata_serverworks low-level driver for Serverworks OSB4/CSB5/CSB6: 0.4.3
kernel module pata_sil680 low-level driver for SI680 PATA: 0.4.9
kernel module pata_sis SCSI low-level driver for SiS ATA: 0.5.2
kernel module pata_sl82c105 low-level driver for SI82c105: 0.3.3
kernel module pata_triflex low-level driver for Compaq Triflex: 0.2.8
kernel module pata_via low-level driver for VIA PATA: 0.3.4
kernel module pdc_adma Pacific Digital Corporation ADMA low-level driver: 1.0
kernel module pm80xx PMC-Sierra PM8001/8006/8081/8088/8089/8074/8076/8077/8070/8072 SAS/SATA controller driver: 0.1.39
kernel module pmcraid PMC Sierra MaxRAID Controller Driver: 1.0.3
kernel module qed QLogic FastLinQ 4xxxx Core Module: 8.37.0.20
kernel module qede QLogic FastLinQ 4xxxx Ethernet Driver: 8.37.0.20
kernel module qedf QLogic FastLinQ 4xxxx FCoE Module: 8.42.3.0
kernel module qedi QLogic FastLinQ 4xxxx iSCSI Module: 8.37.0.20
kernel module qla1280 Qlogic ISP SCSI (qla1x80/qla1x160) driver: 3.27.1
kernel module qla2xxx QLogic Fibre Channel HBA Driver: 10.01.00.19-k
kernel module qla3xxx QLogic ISP3XXX Network Driver v2.03.00-k5 : v2.03.00-k5
kernel module qla4xxx QLogic iSCSI HBA Driver: 5.04.00-k6
kernel module qlcnic QLogic 1/10 GbE Converged/Intelligent Ethernet Driver: 5.3.66
kernel module r6040 RDC R6040 NAPI PCI FastEthernet driver: 0.29 04Jul2016
kernel module rsxx IBM Flash Adapter 900GB Full Height Device Driver: 4.0.3.2516
kernel module s2io : 2.0.26.28
kernel module sata_dwc_460ex DesignWare Cores SATA controller low level driver: 1.3
kernel module sata_mv SCSI low-level driver for Marvell SATA controllers: 1.28
kernel module sata_nv low-level driver for NVIDIA nForce SATA controller: 3.5
kernel module sata_promise Promise ATA TX2/TX4/TX4000 low-level driver: 2.12
kernel module sata_qstor Pacific Digital Corporation QStor SATA low-level driver: 0.09
kernel module sata_sil low-level driver for Silicon Image SATA controller: 2.4
kernel module sata_sis low-level driver for Silicon Integrated Systems SATA controller: 1.0
kernel module sata_svw low-level driver for K2 SATA controller: 2.3
kernel module sata_sx4 Promise SATA low-level driver: 0.12
kernel module sata_uli low-level driver for ULI Electronics SATA controller: 1.3
kernel module sata_via SCSI low-level driver for VIA SATA controllers: 2.6
kernel module sata_vsc low-level driver for Vitesse VSC7174 SATA controller: 2.3
kernel module sfc Solarflare network driver: 4.1
kernel module sfc_falcon Solarflare Falcon network driver: 4.1
kernel module sg SCSI generic (sg) driver: 3.5.36
kernel module sis190 SiS sis190/191 Gigabit Ethernet driver: 1.4
kernel module skge SysKonnect Gigabit Ethernet driver: 1.14
kernel module sky2 Marvell Yukon 2 Gigabit Ethernet driver: 1.30
kernel module slicoss Alacritech non-accelerated SLIC driver: 1.0
kernel module smartpqi Driver for Microsemi Smart Family Controller version 1.2.8-026: 1.2.8-026
kernel module smsc911x : 2008-10-21
kernel module smsc9420 : 1.01
kernel module snic Cisco SCSI NIC Driver: 0.0.1.18

kernel module starfire Adaptec Starfire Ethernet driver: 2.1
kernel module stex Promise Technology SuperTrak EX Controllers: 6.02.0000.01
kernel module sunhme Sun HappyMealEthernet(HME) 10/100baseT ethernet driver: 3.10
kernel module sym53c8xx NCR, Symbios and LSI 8xx and 1010 PCI SCSI adapters: 2.2.3
kernel module tg3 Broadcom Tigon3 ethernet driver: 3.137
kernel module thunder_bgx Cavium Thunder BGX/MAC Driver: 1.0
kernel module thunder_xcv Cavium Thunder RGX/XCV Driver: 1.0
kernel module tpm TPM Driver: 2.0
kernel module tpm_atmel TPM Driver: 2.0
kernel module tpm_crb TPM2 Driver: 0.1
kernel module tpm_i2c_infineon TPM TIS I2C Infineon Driver: 2.2.0
kernel module tpm_infineon Driver for Infineon TPM SLD 9630 TT 1.1 / SLB 9635 TT 1.2: 1.9.2
kernel module tpm_nsc TPM Driver: 2.0
kernel module tpm_st33zp24 ST33ZP24 TPM 1.2 driver: 1.3.0
kernel module tpm_st33zp24_i2c STM TPM 1.2 I2C ST33 Driver: 1.3.0
kernel module tpm_tis TPM Driver: 2.0
kernel module tpm_tis_core TPM Driver: 2.0
kernel module tpm_vtpm_proxy vTPM Driver: 0.1
kernel module tulip Digital 21*4* Tulip ethernet driver: 1.1.15
kernel module typhoon 3Com Typhoon Family (3C990, 3CR990, and variants): 1.0
kernel module ufshcd_core Generic UFS host controller driver Core: 0.2
kernel module virtio_pci virtio-pci: 1
kernel module vmw_pvscsi VMware PVSCSI driver: 1.0.7.0-k
kernel module vmxnet3 VMware vmxnet3 virtual NIC driver: 1.4.17.0-k
kernel module vxlan Driver for VXLAN encapsulated traffic: 0.1
kernel module winbond_840 Winbond W89c840 Ethernet driver: 1.01-e
adduser version: 3.118
apt version: 1.8.2.1
apt-utils version: 1.8.2.1
base-files version: 10.3+deb10u6
base-passwd version: 3.5.46
bash version: 5.0-4
bsdmainutils version: 11.1.2+b1
bsdutils version: 1:2.33.1-0.1
busybox version: 1:1.30.1-4
bzip2 version: 1.0.6-9.2~deb10u1
ca-certificates version: 20190110
coreutils version: 8.30-3
cpio version: 2.12+dfsg-9
cron version: 3.0pl1-134+deb10u1
cryptsetup-bin version: 2:2.1.0-5+deb10u2
curl version: 7.64.0-4+deb10u1
dash version: 0.5.10.2-5
dbus version: 1.12.20-0+deb10u1
debconf version: 1.5.71
debconf-i18n version: 1.5.71
debian-archive-keyring version: 2019.1
debiantutils version: 4.8.6.1
dhcpcd5 version: 7.1.0-2
diffutils version: 1:3.7-3
dirmngr version: 2.2.12-1+deb10u1
dmidecode version: 3.2-1
dmsetup version: 2:1.02.155-3
dosfstools version: 4.1-2

dpkg version: 1.19.7
e2fsprogs version: 1.44.5-1+deb10u3
ethtool version: 1:4.19-1
fdisk version: 2.33.1-0.1
file version: 1:5.35-4+deb10u1
findutils version: 4.6.0+git+20190209-2
gcc-8-base version: 8.3.0-6
gdbm-l10n version: 1.18.1-4
gdisk version: 1.0.3-1.1
gnupg version: 2.2.12-1+deb10u1
gnupg-l10n version: 2.2.12-1+deb10u1
gnupg-utils version: 2.2.12-1+deb10u1
gpg version: 2.2.12-1+deb10u1
gpg-agent version: 2.2.12-1+deb10u1
gpg-wks-client version: 2.2.12-1+deb10u1
gpg-wks-server version: 2.2.12-1+deb10u1
gpgconf version: 2.2.12-1+deb10u1
gpgsm version: 2.2.12-1+deb10u1
gpgv version: 2.2.12-1+deb10u1
grep version: 3.3-1
groff-base version: 1.22.4-3
guile-2.2-libs version: 2.2.4+1-2+deb10u1
gzip version: 1.9-3
hdparm version: 9.58+ds-1
hostname version: 3.21
hwdm version: 21.63-3
ifenslave version: 2.9
ifupdown version: 0.8.35
init version: 1.56+nmu1
init-system-helpers version: 1.56+nmu1
initramfs-tools version: 0.133+deb10u1
initramfs-tools-core version: 0.133+deb10u1
iproute2 version: 4.20.0-2
iptables version: 1.8.2-4
iputils-ping version: 3:20180629-2+deb10u1
irqbalance version: 1.5.0-3
isc-dhcp-client version: 4.4.1-2
isc-dhcp-common version: 4.4.1-2
klibc-utils version: 2.0.6-1
kmod version: 26-1
kpartx version: 0.7.9-3+deb10u1
krb5-locales version: 1.17-3
less version: 487-0.1+b1
libacl1 version: 2.2.53-4
libaio1 version: 0.3.112-3
libapparmor1 version: 2.13.2-10
libapt-inst2.0 version: 1.8.2.1
libapt-pkg5.0 version: 1.8.2.1
libargon2-1 version: 0~20171227-0.2
libassuan0 version: 2.5.2-1
libattr1 version: 1:2.4.48-4
libaudit-common version: 1:2.8.4-3
libaudit1 version: 1:2.8.4-3
libblkid1 version: 2.33.1-0.1

libboost-atomic1.67.0 version: 1.67.0-13+deb10u1
libboost-python1.67.0 version: 1.67.0-13+deb10u1
libboost-system1.67.0 version: 1.67.0-13+deb10u1
libboost-thread1.67.0 version: 1.67.0-13+deb10u1
libbsd0 version: 0.9.1-2
libbz2-1.0 version: 1.0.6-9.2~deb10u1
libc-bin version: 2.28-10
libc6 version: 2.28-10
libcap-ng0 version: 0.7.9-2
libcap2 version: 1:2.25-2
libcap2-bin version: 1:2.25-2
libcom-err2 version: 1.44.5-1+deb10u3
libcryptsetup12 version: 2:2.1.0-5+deb10u2
libcurl4 version: 7.64.0-4+deb10u1
libdb5.3 version: 5.3.28+dfsg1-0.5
libdbus-1-3 version: 1.12.20-0+deb10u1
libdebconfclient0 version: 0.249
libdevmapper1.02.1 version: 2:1.02.155-3
libdns-export1104 version: 1:9.11.5.P4+dfsg-5.1+deb10u2
libedit2 version: 3.1-20181209-1
libelf1 version: 0.176-1.1
libestr0 version: 0.1.10-2.1
libevent-core-2.1-6 version: 2.1.8-stable-4
libevent-pthreads-2.1-6 version: 2.1.8-stable-4
libexpat1 version: 2.2.6-2+deb10u1
libext2fs2 version: 1.44.5-1+deb10u3
libfastjson4 version: 0.99.8-2
libfdisk1 version: 2.33.1-0.1
libffi6 version: 3.2.1-9
libfribidi0 version: 1.0.5-3.1+deb10u1
libgc1c2 version: 1:7.6.4-0.4
libgcc1 version: 1:8.3.0-6
libgcrypt20 version: 1.8.4-5
libgdbm-compat4 version: 1.18.1-4
libgdbm6 version: 1.18.1-4
libglib2.0-0 version: 2.58.3-2+deb10u2
libglib2.0-data version: 2.58.3-2+deb10u2
libgmp10 version: 2:6.1.2+dfsg-4
libgnutls-openssl27 version: 3.6.7-4+deb10u5
libgnutls30 version: 3.6.7-4+deb10u5
libgpg-error0 version: 1.35-1
libgsasl7 version: 1.8.0-8+b2
libgssapi-krb5-2 version: 1.17-3
libhd21 version: 21.63-3
libhogweed4 version: 3.4.1-1
libicu63 version: 63.1-6+deb10u1
libidn11 version: 1.33-2.2
libidn2-0 version: 2.0.5-1+deb10u1
libip4tc0 version: 1.8.2-4
libip6tc0 version: 1.8.2-4
libiptc0 version: 1.8.2-4
libisc-export1100 version: 1:9.11.5.P4+dfsg-5.1+deb10u2
libjson-c3 version: 0.12.1+ds-2+deb10u1
libk5crypto3 version: 1.17-3

libkeyutils1 version: 1.6-6
libklibc version: 2.0.6-1
libkmod2 version: 26-1
libkrb5-3 version: 1.17-3
libkrb5support0 version: 1.17-3
libksba8 version: 1.3.5-2
libkyotocabinet16v5 version: 1.2.76-4.2+b1
libldap-2.4-2 version: 2.4.47+dfsg-3+deb10u2
libldap-common version: 2.4.47+dfsg-3+deb10u2
liblocale-gettext-perl version: 1.07-3+b4
liblognorm5 version: 2.0.5-1
libltdl7 version: 2.4.6-9
liblz4-1 version: 1.8.3-1
liblzma5 version: 5.2.4-1
liblzo2-2 version: 2.10-0.1
libmagic-mgc version: 1:5.35-4+deb10u1
libmagic1 version: 1:5.35-4+deb10u1
libmailutils5 version: 1:3.5-4
libmariadb3 version: 1:10.3.23-0+deb10u1
libmnl0 version: 1.0.4-2
libmount1 version: 2.33.1-0.1
libmpdec2 version: 2.4.2-2
libncurses6 version: 6.1+20181013-2+deb10u2
libncursesw6 version: 6.1+20181013-2+deb10u2
libnetfilter-contrack3 version: 1.0.7-1
libnettle6 version: 3.4.1-1
libnewt0.52 version: 0.52.20-8
libnfnetlink0 version: 1.0.1-3+b1
libnftnl11 version: 1.1.2-2
libnghttp2-14 version: 1.36.0-2+deb10u1
libnph0 version: 1.6-1
libntlm0 version: 1.5-1+deb10u1
libnuma1 version: 2.0.12-1
libopenipmi0 version: 2.0.25-2.1
libopts25 version: 1:5.18.12-4
libp11-kit0 version: 0.23.15-2
libpam-modules version: 1.3.1-5
libpam-modules-bin version: 1.3.1-5
libpam-runtime version: 1.3.1-5
libpam-systemd version: 241-7~deb10u4
libpam0g version: 1.3.1-5
libpcap0.8 version: 1.8.1-6
libpci3 version: 1:3.5.2-1
libpcre3 version: 2:8.39-12
libperl5.28 version: 5.28.1-6+deb10u1
libpopt0 version: 1.16-12
libprocps7 version: 2:3.3.15-2
libpsl5 version: 0.20.2-2
libpython2.7 version: 2.7.16-2+deb10u1
libpython2.7-minimal version: 2.7.16-2+deb10u1
libpython2.7-stdlib version: 2.7.16-2+deb10u1
libpython3.7 version: 3.7.3-2+deb10u2
libpython3.7-minimal version: 3.7.3-2+deb10u2
libpython3.7-stdlib version: 3.7.3-2+deb10u2

libreadline7 version: 7.0-5
librtmp1 version: 2.4+20151223.gitfa8646d.1-2
libsasl2-2 version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules-db version: 2.1.27+dfsg-1+deb10u1
libseccomp2 version: 2.3.3-4
libselinux1 version: 2.8-1+b1
libsemanage-common version: 2.8-2
libsemanage1 version: 2.8-2
libsensors-config version: 1:3.5.0-3
libsensors5 version: 1:3.5.0-3
libsepol1 version: 2.8-1
libsgutils2-2 version: 1.44-1
libslang2 version: 2.3.2-2
libsmartcols1 version: 2.33.1-0.1
libsnmp-base version: 5.7.3+dfsg-5+deb10u1
libsnmp30 version: 5.7.3+dfsg-5+deb10u1
libsqlite3-0 version: 3.27.2-3
libss2 version: 1.44.5-1+deb10u3
libssh2-1 version: 1.8.0-2.1
libssl1.1 version: 1.1.1d-0+deb10u3
libstdc++6 version: 8.3.0-6
libsysfs2 version: 2.1.0+repack-5
libsystemd0 version: 241-7~deb10u4
libtasn1-6 version: 4.13-3
libtext-charwidth-perl version: 0.04-7.1+b1
libtext-iconv-perl version: 1.7-5+b7
libtext-wrapi18n-perl version: 0.06-7.1
libtinfo6 version: 6.1+20181013-2+deb10u2
libuchardet0 version: 0.0.6-3
libudev1 version: 241-7~deb10u4
libunistring2 version: 0.9.10-1
liburcu6 version: 0.10.2-1
libuuid1 version: 2.33.1-0.1
libwrap0 version: 7.6.q-28
libx11-6 version: 2:1.6.7-1+deb10u1
libx11-data version: 2:1.6.7-1+deb10u1
libx86emu2 version: 2.0-1
libxau6 version: 1:1.0.8-1+b2
libxcb1 version: 1.13.1-2
libxdmcp6 version: 1:1.1.2-3
libxext6 version: 2:1.3.3-1+b2
libxml2 version: 2.9.4+dfsg1-7+b3
libxmuu1 version: 2:1.1.2-2+b3
libxtables12 version: 1.8.2-4
libyajl2 version: 2.1.0-3
libzstd1 version: 1.3.8+dfsg-3
linux-base version: 4.6
linux-firmware version: 1.190
login version: 1:4.5-1.1
logrotate version: 3.14.0-4
lsb-base version: 10.2019051400
lsscsi version: 0.30-0.1
mailutils version: 1:3.5-4

mailutils-common version: 1:3.5-4
mariadb-common version: 1:10.3.23-0+deb10u1
mawk version: 1.3.3-17+b3
megacli version: 8.07.14-2+Debian.buster.10
mime-support version: 3.62
mount version: 2.33.1-0.1
multipath-tools version: 0.7.9-3+deb10u1
multipath-tools-boot version: 0.7.9-3+deb10u1
mysql-common version: 5.8+1.0.5
nano version: 3.2-3
ncurses-base version: 6.1+20181013-2+deb10u2
ncurses-bin version: 6.1+20181013-2+deb10u2
ncurses-term version: 6.1+20181013-2+deb10u2
net-tools version: 1.60+git20180626.aebd88e-1
netbase version: 5.6
nload version: 0.7.4-2+b1
ntp version: 1:4.2.8p12+dfsg-4
numactl version: 2.0.12-1
openipmi version: 2.0.25-2.1
openresolv version: 3.8.0-1
openssh-client version: 1:7.9p1-10+deb10u2
openssh-server version: 1:7.9p1-10+deb10u2
openssh-sftp-server version: 1:7.9p1-10+deb10u2
openssl version: 1.1.1d-0+deb10u3
passwd version: 1:4.5-1.1
perl version: 5.28.1-6+deb10u1
perl-base version: 5.28.1-6+deb10u1
perl-modules-5.28 version: 5.28.1-6+deb10u1
pigz version: 2.4-1
pinentry-curses version: 1.1.0-2
powermgmt-base version: 1.34
procps version: 2:3.3.15-2
publicsuffix version: 20190415.1030-1
python3.7 version: 3.7.3-2+deb10u2
python3.7-minimal version: 3.7.3-2+deb10u2
qemu-guest-agent version: 1:3.1+dfsg-8+deb10u8
readline-common version: 7.0-5
rsyslog version: 8.1901.0-1
runit-helper version: 2.8.6
sdparm version: 1.10-1
sed version: 4.7-1
sensible-utils version: 0.0.12
sg3-utils version: 1.44-1
sg3-utils-udev version: 1.44-1
shared-mime-info version: 1.10-1
smartmontools version: 6.6-1
smp-utils version: 0.98-2
snmp version: 5.7.3+dfsg-5+deb10u1
snmpd version: 5.7.3+dfsg-5+deb10u1
snmp version: 1:4.2.8p12+dfsg-4
ssmtp version: 2.64-8.1
sysstat version: 12.0.3-2
systemd version: 241-7~deb10u4
systemd-sysv version: 241-7~deb10u4

sysvinit-utils version: 2.93-8
tar version: 1.30+dfsg-6
tasksel version: 3.53
tasksel-data version: 3.53
tcpdump version: 4.9.3-1~deb10u1
tofrodos version: 1.7.13+ds-4
traceroute version: 1:2.1.0-2
tzdata version: 2020a-0+deb10u1
ucf version: 3.0038+nmu1
udev version: 241-7~deb10u4
util-linux version: 2.33.1-0.1
vim-common version: 2:8.1.0875-5
vim-tiny version: 2:8.1.0875-5
whiptail version: 0.52.20-8
xauth version: 1:1.0.10-1
xdg-user-dirs version: 0.17-2
xxd version: 2:8.1.0875-5
xz-utils version: 5.2.4-1
zlib1g version: 1:1.2.11.dfsg-1
linux-firmware: 1.190
megacli: 8.07.14-2+Debian.buster.10
ssmtp: 2.64-8.1

Swarm Storage 11 Releases

- [Swarm Storage 11.3 Release](#)
- [Swarm Storage 11.2 Release](#)
- [Swarm Storage 11.1 Release](#)
- [Swarm Storage 11.0 Release](#)

Swarm Storage 11.3 Release

- [New Features](#)
- [Additional Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Known Issues](#)
- [Upgrading Swarm](#)

New Features

Performance – This release of Swarm Storage enhances both memory management and cluster performance:

- Swarm cluster startup has been optimized to guarantee the fastest sequencing. Now volume mounting must complete and the persistent settings must be processed before any needed recovery activities can commence. (SWAR-8911)
- Swarm nodes shut down faster, allowing for quicker rebooting of Swarm clusters. (SWAR-8891)
- Swarm nodes with limited physical memory can now respond better under high client loads. (SWAR-8870)
- Swarm’s memory management has been improved, which enables higher loads for client writes. (SWAR-8816)

Stability – This release also includes changes that improve Swarm stability and administration:

- Better handling of newly added hotplug volumes results in clients receiving fewer 503 Service Unavailable responses. (SWAR-8887)
- HP cycles now cleanse all traces of removed volumes from the cluster, greatly reducing the chance that recovery can be started erroneously for a volume already recovered. (SWAR-8836)
- Reworking of cluster operations has reduced spurious "Cannot contact node" announcements during maintenance rebooting of multiple nodes. (SWAR-8848)
- When secure logging (`security.secureLogging`) is enabled, Swarm removes more sensitive information from AUDIT-level messages. (SWAR-8790)

Additional Changes

These items are other changes, including those that come from testing and user feedback.

- **OSS Versions** – See [Third-Party Components for 11.3](#) for the complete listing of packages and versions.
- **Fixed in 11.3.0**
 - Drive light plug-in control is restored for hardware in mpt3sas enclosures, including Western Digital Ultrastar Serv60. (SWAR-8934)
 - For some feed statistics, feed accounting resets and requires a reboot to correct the statistic. (SWAR-8854)

Upgrade Impacts

Use the supported versions of Swarm components if running an older version of Elasticsearch:

Elasticsearch 6.8.6	Swarm Storage 11.1 - 11.3	Gateway 6.3	SwarmFS 2.4	Recommended configuration.
Elasticsearch 5.6.12	Swarm Storage 10.0 - 11.3	Gateway 6.0 - 6.3	SwarmFS 2.4	Plan to migrate to Elasticsearch 6 . Support for earlier versions is ending.

Elasticsearch 2.3.3	Swarm Storage 9.6 - 11.3	Gateway 5.4	SwarmFS 2.1
---------------------	--------------------------	-------------	-------------

These items are changes to the product function that may require operational or development changes for integrated applications. Address the upgrade impacts for each of the versions since the version currently running:

Impacts for 11.3

- **Upgrading Elasticsearch** – Use Elasticsearch 5.6.12/2.3.3 with Storage 11 if moving to ES 6 immediately is not possible, but start the migration now (see [Migrating from Older Elasticsearch](#)). Support for ES 5.6.12/2.3.3 ends in a future release, and testing for 2.3.3 with Swarm 11 is discontinued. *Important:* Always upgrade Swarm Search and Metrics at the same time ES is upgrade . Do not run an ES 5 Search or Metrics Curator against ES 6.
- **Rolling upgrade** – During a rolling upgrade from a version older than 11.1, the mixed state in Swarm versions among nodes may cause errors in the Swarm UI (and in management API calls). Use the legacy Admin Console (port 90) to monitor the rolling upgrade. (SWAR-8716)
- **Settings changes** – The setting `health.parallelWriteTimeout`, which was disabled by default, now defaults to 1 month. It sets when to time out an uncompleted multipart upload, triggering clean up of the unused parts. Do not disable (0) if using SwarmFS. (SWAR-8902)
- **Encryption-at-rest** –If upgrading from Swarm 11.0 or earlier and encryption-at-rest is used, contact DataCore Support to verify a roll back to the prior version is possible, if needed. (SWAR-8941)
- **Differences in `scsp.forceLegacyNonce` configuration** depending on the version being upgraded from (SWAR-9020):
- **If currently running a Swarm Storage version prior to 11.1**, and upgrading to 11.1, 11.2, 11.3, 12.0 or 12.1:

Before upgrading, set `scsp.forceLegacyNonce=true` in the `node.cfg` file. After the upgrade, when the cluster is fully up, update `scsp.forceLegacyNonce=false` using `swarmctl` and change `scsp.forceLegacyNonce=false` in the `node.cfg` file.

If currently running a Swarm Storage version 11.1, 11.2, 11.3, 12.0 or 12.1 and upgrading to another version from that list:

Before upgrading, verify the `scsp.forceLegacyNonce=false` is in the `node.cfg` file and verify using `swarmctl` that `scsp.forceLegacyNonce=false` in the cluster.

Use swarmctl to check or change settings

Use `'swarmctl -C scsp.forceLegacyNonce'` to check the value of `scsp.forceLegacyNonce`.

Use `'swarmctl -C scsp.forceLegacyNonce -V False'` to set the value to false.

For more details, see <https://support.cloud.caringo.com/tools/Tech-Support-Scripts-Bundle-swarmctl.pdf>.

Impacts for 11.2

- **Upgrading Elasticsearch** – Elasticsearch 5.6.12/2.3.3 may be used with Storage 11 if move to ES 6 cannot be performed immediately, but start the migration now (see [Migrating from Older Elasticsearch](#)). Support for ES 5.6.12/2.3.3 ends in a future release, and testing for 2.3.3 with Swarm 11 is discontinued. *Important:* Always upgrade Swarm Search and Metrics at the same time upgrading ES. Do not run an ES 5 Search or Metrics Curator against ES 6.
- **Rolling upgrade** – During a rolling upgrade, the mixed state in Swarm versions among nodes may cause errors in the Swarm UI (and in management API calls). Use the legacy Admin Console (port 90) to monitor the rolling upgrade. (SWAR-8716)
- **Settings changes** – These settings are new with this release:

- `scsp.defaultFeedSendTimeout`, (default 30 seconds) a non-persisted node-level setting that sets the timeout on a feed SEND request, if the `timeout=true` query argument is provided. (SWAR-8441).
- `chassis.name`, (default blank), a node-level setting that stores a user-defined chassis name. (SWAR-8823)
- **Differences in `scsp.forceLegacyNonce` configuration** depending on the version upgrading from (SWAR-9020):
- **If currently running a Swarm Storage version prior to 11.1**, and upgrading to 11.1, 11.2, 11.3, 12.0 or 12.1:

Before upgrading, set `scsp.forceLegacyNonce=true` in the `node.cfg` file. After the upgrade, when the cluster is fully up, update `scsp.forceLegacyNonce=false` using `swarmctl` and change `scsp.forceLegacyNonce=false` in the `node.cfg` file.

If currently running a Swarm Storage version 11.1, 11.2, 11.3, 12.0 or 12.1 and upgrading to another version from that list:

Before upgrading, verify `scsp.forceLegacyNonce=false` is in the `node.cfg` file and verify using `swarmctl` that `scsp.forceLegacyNonce=false` in the cluster.

Use `swarmctl` to check or change settings

Use `'swarmctl -C scsp.forceLegacyNonce'` to check the value of `scsp.forceLegacyNonce`.

Use `'swarmctl -C scsp.forceLegacyNonce -V False'` to set the value to false.

For more details, see <https://support.cloud.caringo.com/tools/Tech-Support-Scripts-Bundle-swarmctl.pdf>.

Impacts for 11.1

- **Upgrading Elasticsearch** – Use Elasticsearch 5.6.12/2.3.3 with Storage 11.1 if moving to ES 6 immediately is not possible, but start the migration now (see [Migrating from Older Elasticsearch](#)). Support for ES 5.6.12/2.3.3 ends in a future release, and testing for 2.3.3 with Swarm 11 is discontinued. *Important:* Always upgrade Swarm Search and Metrics at the same time ES is upgraded. Do not run an ES 5 Search or Metrics Curator against ES 6.
- **Swarm Search and Metrics** – This release includes new versions of Swarm Search and Metrics RPMs. Both require Python 3 to be installed on the ES servers they run on.
 - For Swarm Metrics on RHEL/CentOS 7.7, first install this dependency: `yum install epel-release`
- **Python 3** – Install Python 3 if is not automatically installed with RHEL/CentOS 7.
- **Propagate Delete Removed** – For [replication and S3 backup feeds](#), the Propagate Deletes option is removed from the legacy Admin Console and the Management API (`propagateDeletes`, `nodeletes` fields). (SWAR-8609, SWAR-8615)
- **Swarm Configuration** – Run the [Storage Settings Checker](#) before upgrading to this version, to identify configuration issues.
 - The Storage Settings Checker now requires Python 3 to be installed. (SWAR-8742)
 - `crier.deadVolumeWall` has been unpublished for reimplement. (SWAR-8640)
- **S3 Backup Restore** – The S3 Backup Restore Tool has been migrated to Python 3.6. If the tool is installed, uninstall it and [install the new version](#). (SWAR-8703)
- **Upgrade Process** – During the upgrade to 11.1, it may not be possible to monitor the cluster via the Swarm UI. Workaround: Use the legacy Admin Console (port 90) during upgrade. (SWAR-8716)
- **Differences in `scsp.forceLegacyNonce` configuration** depending on the version being upgraded from (SWAR-9020):
- **If currently running a Swarm Storage version prior to 11.1**, and upgrading to 11.1, 11.2, 11.3, 12.0 or 12.1:

Before upgrading, set `scsp.forceLegacyNonce=true` in the `node.cfg` file. After the upgrade, when the cluster is fully up, update `scsp.forceLegacyNonce=false` using `swarmctl` and change `scsp.forceLegacyNonce=false` in the `node.cfg` file.

If currently running a Swarm Storage version 11.1, 11.2, 11.3, 12.0 or 12.1 and upgrading to another version from that list:

Before upgrading, verify `scsp.forceLegacyNonce=false` is in the `node.cfg` file and verify using `swarmctl` that `scsp.forceLegacyNonce=false` in the cluster.

Use swarmctl to check or change settings

Use `'swarmctl -C scsp.forceLegacyNonce'` to check the value of `scsp.forceLegacyNonce`.

Use `'swarmctl -C scsp.forceLegacyNonce -V False'` to set the value to `false`.

For more details, see <https://support.cloud.caringo.com/tools/Tech-Support-Scripts-Bundle-swarmctl.pdf>.

Impacts for 11.0

- **Upgrading Elasticsearch** – You may use Elasticsearch 2.3.3 with Storage 11.0 if you cannot move to 5.6 now, but plan the migration immediately (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release, and testing with Swarm 11 is discontinuing.
- **Propagate Deletes Deprecated** – The option to disable *Propagate Deletes* on [replication feeds](#) is deprecated; use [Object Versioning](#) to preserve deleted content. Do not disable Propagate Deletes when versioning is enabled or when defining an S3 Backup. (SWAR-8609)
- **Configuration Settings** – Run the [Storage Settings Checker](#) to identify these and other configuration issues.
 - Changed settings:
 - `ec.segmentConsolidationFrequency` (`ecSegmentConsolidationFrequency` in SNMP) has an improved default (10), which you must apply to your cluster when you upgrade. (SWAR-8483)
 - `cluster.name` is now required. Add it to the `cluster.cfg` file. (SWAR-8466).
 - `metrics.nodeExporterFrequency` (`metricsExporterFrequency` in SNMP) is now a persisted cluster setting. (SWAR-8467).
 - Removed settings:
 - `chassis.processes` is allowed but is ignored.
 - Numerous settings are now promoted to *cluster-level* (versus node-level) scope, so you can manage them via **Settings > Cluster** in the Swarm UI (SWAR-8457):
 - `console.expiryErrInterval`
 - `console.expiryWarnInterval`
 - `console.indexErrorLevel`
 - `console.indexWarningLevel`
 - `console.port`
 - `console.reportStyleUrl`
 - `console.spaceErrorLevel`
 - `console.spaceWarnLevel`
 - `console.styleUrl`
 - `feeds.retry`
 - `feeds.statsReportInterval`
 - `health.parallelWriteTimeout`
 - `log.obscureUUIDs`
 - `metrics.enableNodeExporter`
 - `network.dnsDomain`
 - `network.dnsServers`
 - `network.icmpAcceptRedirects`

- `network.igmpVersion`
- `network.mtu`
- `startup.certificates`

Impacts for 10.2

- **Upgrading Elasticsearch** – You may continue to use Elasticsearch 2.3.3 with Storage 10.2 until you are able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release. the upgrade to Elasticsearch 5.6 must be completed before [upgrading to Gateway 6.0](#).
- **Configuration Settings** – Run the [Storage Settings Checker](#) before any Swarm 10 upgrade to identify configuration issues. Note these changes:
 - `ec.protectionLevel` is now persisted. (SWAR-8231)
 - `index.ovMinNodes=3` is the new default for the overlay index, in support of Swarm 10's new architecture. To keep your overlay index operational, set this new value in your cluster, through the UI or by SNMP (`overlayMinNodes`). (SWAR-8278)
 - `metrics.enableNodeExporter` can be set to True, which enables the Prometheus Node Exporter on that node. (SWAR-8408, SWAR-8578)
 - `metrics.nodeExporterFrequency`, a new dynamic setting, sets how frequently to refresh Swarm-specific Prometheus metrics in Elasticsearch; it defaults to 0, which disables this export. (SWAR-8408).

Impacts for 10.1

- **Upgrading Elasticsearch** – Continue to use Elasticsearch 2.3.3 with Storage 10.1 until able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release. Complete the upgrade to Elasticsearch 5.6 before upgrading to Gateway 6.0.
- **Configuration Settings** – Run the [Storage Settings Checker](#) before any Swarm 10 upgrade to identify configuration issues.
 - `metrics.enableNodeExporter=true` enables Swarm to run the Prometheus node exporter on port 9100. (SWAR-8170)
- **IP address update delay** – When upgrading from Swarm 9 to the new architecture of Swarm 10, note the "ghosts" of previously used IP addresses may appear in the Storage UI; these resolve within 4 days. (SWAR-8351)
- **Update MIBs on CSN** – Before upgrading to Storage 10.x, the MIBs on the CSN must be updated. From the Swarm Support tools bundle, run the `platform-update-mibs.sh` script. (CSN-1872)

Impacts for 10.0

- **Upgrading Elasticsearch** – You may continue to use Elasticsearch 2.3.3 with Storage 10.0 until you are able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release.
- **Configuration Settings** – Run the [Storage Settings Checker](#) to identify these and other configuration issues.
 - Changes for the new single-IP dense architecture:
 - `network.ipAddress` – multiple IP addresses now disallowed
 - `chassis.processes` – removed; multi-server configurations are no longer supported
 - `ec.protectionLevel` – new value "volume"
 - `ec.subclusterLossTolerance` – removed
 - Changes for security (see next section)
 - `security.administrators`, `security.operators` – removed 'snmp' user
 - `snmp.rwCommunity`, `snmp.roCommunity` – new settings for 'snmp' user
 - `startup.certificates` – new setting to hold any and all public keys
 - New settings:
 - `disk.atimeEnabled`
 - `health.parallelWriteTimeout`

- `search.pathDelimiter`
- **Required SNMP security change** – Remove the `snmp` key from the `security.administrators` setting, and update `snmp.rwCommunity` with its value. Nodes that contain only the `snmp` key in the `security.administrators` setting does not boot. If you changed the default value of the `snmp` key in the `security.operators` setting, update `snmp.roCommunity` with that value and then remove the `snmp` key from `security.operators`. In the `security.operators` setting, 'snmp' is a reserved key, and it cannot be an authorized console operator name. (SWAR-8097)
- **EC protection**
 - *Best practice:* Use `ec.protectionLevel=node`, which distributes segments across the cluster's physical/virtual machines. Do not use `ec.protectionLevel=subcluster` unless you already have subclusters defined and are sure the specified EC encoding is supported. A new level, `ec.protectionLevel=volume`, allows EC writes to succeed if you have a small cluster with fewer than $(k+p)/p$ nodes. (Swarm always seeks the highest protection possible for EC segments, regardless of the level you set.)
 - Optimize hardware for EC by verifying there are more than $k+p$ subclusters/nodes (as set by `ec.protectionLevel`); for example, with `policy.ecEncoding=5:2`, you need at least 8 subclusters/nodes. When Swarm cannot distribute EC segments adequately for protection, EC writes can fail despite ample free space. (SWAR-7985)
 - Setting `ec.protectionLevel=subcluster` without creating subclusters (defining `node.subcluster` across sets of nodes) causes a critical error and lowers the protection level to 'node'. (SWAR-8175)
- **Small clusters** – Verify the following settings if using 10 or fewer Swarm nodes. Do not use fewer than 3 in production. *Important:* If you need to change any, do so *before* upgrading to Swarm 10.
 - **policy.replicas** – The `min` and `default` values for numbers of replicas to keep in your cluster must not exceed your number of nodes. For example, a 3-node cluster may have only `min=2` or `min=3`.
 - **EC encoding and protection** – For EC encoding, verify you have enough nodes to support the cluster's encoding (`policy.ecEncoding`). For EC writes to succeed with fewer than $(k+p)/p$ nodes, use the new level, `ec.protectionLevel=volume`.
 - *Best practice:* Keep at least one physical machine in your cluster beyond the minimum number needed. This allows for one machine to be down for maintenance without compromising the constraint.
- **"Cluster in a box"** – Swarm supports a "cluster in a box" configuration as long as that box is running a virtual machine host and Swarm instances are running in 3 or more VMs. Each VM boots separately and has its own IP address. Follow the recommendations for small clusters, substituting VMs for nodes. If you have two physical machines, use the "cluster in a box" configuration, but move to direct booting of Swarm with 3 or more.
- **Offline node status** – Because Swarm 10's new architecture reduces the number of IP addresses in your storage cluster, you may see the old IPs and subclusters reporting as **Offline** nodes until they timeout in 4 days (`crier.forgetOfflineInterval`), which is expected.

Watch Items and Known Issues

The following watch items are known:

- Volumes newly formatted in Swarm 11.1, 11.2, or 11.3 to use encryption-at-rest cannot be downgraded to Swarm 11.0 or earlier without a special procedure to prevent data loss. Contact DataCore Support before any such downgrade with encrypted volumes. (SWAR-8941)
- Infrequent WARNING messages, "Node/Volume entry not published due to lock contention (...); action will be retried," may appear in logs. Unless they are frequent, they may be ignored. (SWAR-8802)
- If a node mounts an encrypted volume that is missing the encryption key in the configuration, the node fails to mount all disks in the node. (SWAR-8762)
- S3 Backup feeds do not backup logical objects greater than 5 GB. (SWAR-8554)
- When restarting a cluster of virtual machines that are UEFI-booted (versus legacy BIOS), the chassis shut down but do not come back up. (SWAR-8054)
- With multipath-enabled hardware, the Swarm console Disk Volume Menu may erroneously show too many disks, having multiplied the actual disks in use by the number of possible paths to them. (SWAR-7248)

These are standing operational limitations:

- If downgrading from Swarm 11.0, CRITICAL errors may appear on the feeds. To stop the errors, edit the existing feed definition names via the Swarm UI or legacy Admin Console. (SWAR-8543)
- If the Elasticsearch cluster is wiped, the Storage UI shows no NFS config. Contact DataCore Support for help repopulating the SwarmFS config information. (SWAR-8007)
- If a bucket is deleted, any incomplete multipart upload into that bucket leaves the parts (unnamed streams) in the domain. To find and delete them, use the s3cmd utility (search the Support site for "s3cmd" for guidance). (SWAR-7690)
- Removing subcluster assignments in the CSN UI creates invalid config parameters preventing the unassigned nodes from booting. (SWAR-7675)
- Logs showed the error "FEEDS WARNING: calcFeedInfo(etag=xxx) cannot find domain xxx, which is needed for a domains-specific replication feed". The root cause is fixed; if such warnings are received, contact DataCore Support so the issue can be resolved. (SWAR-7556)
- If a feed is subject to a prolonged outage, a node reboot may be required for it to resume progress after the outage is cleared. If progress is not resolved after the reboot, contact DataCore Support. This has been resolved in 12.1.0 (SWAR-9062)
- If Elasticsearch 6.8.6 blocks an index due to low disk space, this needs to be issued against each index (`index_*`, `csmeter*`, `metrics*`) in the `read_only_allow_delete` state. This is no longer an issue after upgrading to Swarm 12 / Elasticsearch 7 as it automatically unblocks when disk space frees up. (SWAR-8944)

```
curl -i -XPUT "<ESSERVERIP>:9200/<INDEXNAME>/_settings" -d '{"index.blocks.read_only_allow_delete" : null}' -H "Content-Type: application/json"
```

Upgrading Swarm

Note these installation issues when upgrading Swarm:

- The `elasticsearch-curator` package may show an error during an upgrade, which is a known curator issue. Workaround: Reinstall the curator: `yum reinstall elasticsearch-curator` (SWAR-7439)
- Do not install the Swarm Search RPM before installing Java. If Gateway startup fails with "Caringo script plugin is missing from indexer nodes", uninstall and reinstall the Swarm Search RPM. (SWAR-7688)

Proceed to [How to Upgrade Swarm](#) to upgrade Swarm 9 or higher.



Important

Contact DataCore Support for guidance if needing to migrate from Swarm 8.x or earlier.

Third-Party Components for 11.3

zope.interface version: 4.7.1
ipaddress version: 1.0.23
cryptography version: 2.8
pyOpenSSL version: 19.1.0
service_identity version: 18.1.0
incremental version: 17.5.0
Twisted[tls] version: 19.10.0
pyutil version: 3.3.0
python-dateutil version: 2.8.1
Werkzeug version: 0.16.0
klein version: 19.6.0
requests version: 2.21.0
zfec version: 1.5.3
yajl-py version: 2.1.2
certifi version: *latest as of 2020-08-11*
pyratemp version: 0.3.2
Newt version: 0.52.20
Prometheus node_exporter version: 0.18.1
libpng version: 1.2.8
LILO version: 22.7.1
Operating system: Debian GNU/Linux 10 (buster)
Linux kernel: 4.19.84
kernel module 3w_9xxx 3ware 9000 Storage Controller Linux Driver: 2.26.02.014
kernel module 3w_sas LSI 3ware SAS/SATA-RAID Linux Driver: 3.26.02.000
kernel module 3w_xxxx 3ware Storage Controller Linux Driver: 1.26.02.003
kernel module 8021q : 1.8
kernel module 8139cp RealTek RTL-8139C+ series 10/100 PCI Ethernet driver: 1.3
kernel module 8139too RealTek RTL-8139 Fast Ethernet driver: 0.9.28
kernel module aacraid Dell PERC2, 2/Si, 3/Si, 3/Di, Adaptec Advanced Raid Products, HP NetRAID-4M, IBM ServeRAID & ICP SCSI driver: 1.2.1
[50877]-custom
kernel module acard_ahci ACard AHCI SATA low-level driver: 1.0
kernel module ad7418 AD7416/17/18 driver: 0.4
kernel module ahci AHCI SATA low-level driver: 3.0
kernel module aic79xx Adaptec AIC790X U320 SCSI Host Bus Adapter driver: 3.0
kernel module aic7xxx Adaptec AIC77XX/78XX SCSI Host Bus Adapter driver: 7.0
kernel module aic94xx Adaptec aic94xx SAS/SATA driver: 1.0.3
kernel module am53c974 AM53C974 SCSI driver: 1.00
kernel module amd_xgbe AMD 10 Gigabit Ethernet Driver: 1.0.3
kernel module arcmsr Areca ARC11xx/12xx/16xx/188x SAS/SATA RAID Controller Driver: v1.40.00.09-20180709
kernel module ata_generic low-level driver for generic ATA: 0.2.15
kernel module ata_piix SCSI low-level driver for Intel PIIX/ICH ATA controllers: 2.13
kernel module atl1 Atheros L1 Gigabit Ethernet Driver: 2.1.3
kernel module atl1c Qualcomm Atheros 100/1000M Ethernet Network Driver: 1.0.1.1-NAPI
kernel module atl1e Atheros 1000M Ethernet Network Driver: 1.0.0.7-NAPI
kernel module atl2 Atheros Fast Ethernet Network Driver: 2.2.3
kernel module atlantic aQuantia Corporation(R) Network Driver: 2.0.3.0-kern
kernel module atxp1 System voltages control via Attansic ATXP1: 0.6.3
kernel module b44 Broadcom 44xx/47xx 10/100 PCI ethernet driver: 2.0
kernel module be2iscsi Emulex OneConnectOpen-iSCSI Driver version11.4.0.1 Driver 11.4.0.1: 11.4.0.1
kernel module be2net Emulex OneConnect NIC Driver 12.0.0.0: 12.0.0.0

kernel module bfa QLogic BR-series Fibre Channel HBA Driver fcpim: 3.2.25.1
kernel module bna QLogic BR-series 10G PCIe Ethernet driver: 3.2.25.1
kernel module bnx2 QLogic BCM5706/5708/5709/5716 Driver: 2.2.6
kernel module bnx2fc QLogic FCoE Driver: 2.11.8
kernel module bnx2i QLogic NetXtreme II BCM5706/5708/5709/57710/57711/57712/57800/57810/57840 iSCSI Driver: 2.7.10.1
kernel module bnx2x QLogic BCM57710/57711/57711E/57712/57712_MF/57800/57800_MF/57810/57810_MF/57840/57840_MF Driver: 1.712.30-0
kernel module bnxt_en Broadcom BCM573xx network driver: 1.9.2
kernel module bonding Ethernet Channel Bonding Driver, v3.7.1: 3.7.1
kernel module cnic QLogic cnic Driver: 2.5.22
kernel module csiostor Chelsio FCoE driver: 1.0.0-ko
kernel module cxgb3 Chelsio T3 Network Driver: 1.1.5-ko
kernel module cxgb3i Chelsio T3 iSCSI Driver: 2.0.1-ko
kernel module cxgb4 Chelsio T4/T5/T6 Network Driver: 2.0.0-ko
kernel module cxgb4i Chelsio T4-T6 iSCSI Driver: 0.9.5-ko
kernel module cxgb4vf Chelsio T4/T5/T6 Virtual Function (VF) Network Driver: 2.0.0-ko
kernel module dca : 1.12.1
kernel module dcdbas Dell Systems Management Base Driver (version 5.6.0-3.2): 5.6.0-3.2
kernel module de2104x Intel/Digital 21040/1 series PCI Ethernet driver: 0.7
kernel module dmfe Davicom DM910X fast ethernet driver: 1.36.4
kernel module e100 Intel(R) PRO/100 Network Driver: 3.5.24-k2-NAPI
kernel module e1000 Intel(R) PRO/1000 Network Driver: 7.3.21-k8-NAPI
kernel module e1000e Intel(R) PRO/1000 Network Driver: 3.2.6-k
kernel module eeprom_93cx6 EEPROM 93cx6 chip driver: 1.0
kernel module efivars sysfs interface to EFI Variables: 0.08
kernel module ena Elastic Network Adapter (ENA): 1.5.0K
kernel module enic Cisco VIC Ethernet NIC Driver: 2.3.0.53
kernel module esas2r esas2r: 1.00
kernel module esp_scsi ESP SCSI driver core: 2.000
kernel module fm10k Intel(R) Ethernet Switch Host Interface Driver: 0.23.4-k
kernel module fnic Cisco FCoE HBA Driver: 1.6.0.34
kernel module hpsa Driver for HP Smart Array Controller version 3.4.20-125: 3.4.20-125
kernel module i40e Intel(R) 40-10 Gigabit Ethernet Connection Network Driver: 2.7.29
kernel module i40e Intel(R) Ethernet Connection XL710 Network Driver: 2.3.2-k
kernel module i40evf Intel(R) XL710 X710 Virtual Function Network Driver: 3.2.2-k
kernel module ice Intel(R) Ethernet Connection E800 Series Linux Driver: ice-0.7.0-k
kernel module igb Intel(R) Gigabit Ethernet Network Driver: 5.4.0-k
kernel module igbvf Intel(R) Gigabit Virtual Function Network Driver: 2.4.0-k
kernel module ioatdma : 4.00
kernel module ipmi_msghandler Incoming and outgoing message routing for an IPMI interface.: 39.2
kernel module ipr IBM Power RAID SCSI Adapter Driver: 2.6.4
kernel module ips IBM ServeRAID Adapter Driver 7.12.05: 7.12.05
kernel module iscsi : 1.2.0
kernel module ixgb Intel(R) PRO/10GbE Network Driver: 1.0.135-k2-NAPI
kernel module ixgbe Intel(R) 10 Gigabit PCI Express Network Driver: 5.1.0-k
kernel module ixgbe Intel(R) 10GbE PCI Express Linux Network Driver: 5.5.5
kernel module ixgbevfn Intel(R) 10 Gigabit Virtual Function Network Driver: 4.1.0-k
kernel module jme JMicron JMC2x0 PCI Express Ethernet driver: 1.0.8
kernel module libcxgb Chelsio common library: 1.0.0-ko
kernel module libcxgbi Chelsio iSCSI driver library: 0.9.1-ko
kernel module liquidio Cavium LiquidIO Intelligent Server Adapter Driver: 1.7.2
kernel module liquidio_vf Cavium LiquidIO Intelligent Server Adapter Virtual Function Driver: 1.7.2
kernel module lpfc Emulex LightPulse Fibre Channel SCSI driver 12.0.0.6: 0
kernel module megaraid LSI Logic MegaRAID legacy driver: 2.00.4

kernel module megaraid_mbox LSI Logic MegaRAID Mailbox Driver: 2.20.5.1
kernel module megaraid_mm LSI Logic Management Module: 2.20.2.7
kernel module megaraid_sas Avago MegaRAID SAS Driver: 07.706.03.00-rc1
kernel module mlx4_core Mellanox ConnectX HCA low-level driver: 4.0-0
kernel module mlx4_en Mellanox ConnectX HCA Ethernet driver: 4.0-0
kernel module mlx5_core Mellanox 5th generation network adapters (ConnectX series) core driver: 5.0-0
kernel module mpt3sas LSI MPT Fusion SAS 3.0 Device Driver: 26.100.00.00
kernel module mptbase Fusion MPT base driver: 3.04.20
kernel module mptctl Fusion MPT misc device (ioctl) driver: 3.04.20
kernel module mptfc Fusion MPT FC Host driver: 3.04.20
kernel module mptsas Fusion MPT SAS Host driver: 3.04.20
kernel module mptscsih Fusion MPT SCSI Host driver: 3.04.20
kernel module mptspi Fusion MPT SPI Host driver: 3.04.20
kernel module mtip32xx Micron RealSSD PCIe Block Driver: 1.3.1
kernel module mvsas Marvell 88SE6440 SAS/SATA controller driver: 0.8.16
kernel module myri10ge Myricom 10G driver (10GbE): 1.5.3-1.534
kernel module netxen_nic QLogic/NetXen (1/10) GbE Intelligent Ethernet Driver: 4.0.82
kernel module nfp The Netronome Flow Processor (NFP) driver.: 4.19.84
kernel module nicpf Cavium Thunder NIC Physical Function Driver: 1.0
kernel module nicvf Cavium Thunder NIC Virtual Function Driver: 1.0
kernel module niu NIU ethernet driver: 1.1
kernel module nvme : 1.0
kernel module nvme_core : 1.0
kernel module pata_acpi SCSI low-level driver for ATA in ACPI mode: 0.2.3
kernel module pata_ali low-level driver for ALi PATA: 0.7.8
kernel module pata_amd low-level driver for AMD and Nvidia PATA IDE: 0.4.1
kernel module pata_artop SCSI low-level driver for ARTOP PATA: 0.4.6
kernel module pata_atiixp low-level driver for ATI IXP200/300/400: 0.4.6
kernel module pata_atp867x low level driver for Artop/Acard 867x ATA controller: 0.7.5
kernel module pata_cmd64x low-level driver for CMD64x series PATA controllers: 0.2.18
kernel module pata_efar SCSI low-level driver for EFAR PIIX clones: 0.4.5
kernel module pata_hpt366 low-level driver for the Highpoint HPT366/368: 0.6.11
kernel module pata_hpt37x low-level driver for the Highpoint HPT37x/30x: 0.6.23
kernel module pata_hpt3x2n low-level driver for the Highpoint HPT3xxN: 0.3.15
kernel module pata_hpt3x3 low-level driver for the Highpoint HPT343/363: 0.6.1
kernel module pata_it821x low-level driver for the IT8211/IT8212 IDE RAID controller: 0.4.2
kernel module pata_jmicron SCSI low-level driver for Jmicron PATA ports: 0.1.5
kernel module pata_marvell SCSI low-level driver for Marvell ATA in legacy mode: 0.1.6
kernel module pata_mpiix low-level driver for Intel MPIIX: 0.7.7
kernel module pata_netcell SCSI low-level driver for Netcell PATA RAID: 0.1.7
kernel module pata_ninja32 low-level driver for Ninja32 ATA: 0.1.5
kernel module pata_ns87410 low-level driver for Nat Semi 87410: 0.4.6
kernel module pata_ns87415 ATA low-level driver for NS87415 controllers: 0.0.1
kernel module pata_oldpiix SCSI low-level driver for early PIIX series controllers: 0.5.5
kernel module pata_pdc2027x libata driver module for Promise PDC20268 to PDC20277: 1.0
kernel module pata_pdc202xx_old low-level driver for Promise 2024x and 20262-20267: 0.4.3
kernel module pata_platform low-level driver for platform device ATA: 1.2
kernel module pata_rdc SCSI low-level driver for RDC PATA controllers: 0.01
kernel module pata_rz1000 low-level driver for RZ1000 PCI ATA: 0.2.4
kernel module pata_sch SCSI low-level driver for Intel SCH PATA controllers: 0.2
kernel module pata_serverworks low-level driver for Serverworks OSB4/CSB5/CSB6: 0.4.3
kernel module pata_sil680 low-level driver for SI680 PATA: 0.4.9
kernel module pata_sis SCSI low-level driver for SiS ATA: 0.5.2
kernel module pata_sl82c105 low-level driver for SI82c105: 0.3.3

kernel module pata_triflex low-level driver for Compaq Triflex: 0.2.8
 kernel module pata_via low-level driver for VIA PATA: 0.3.4
 kernel module pdc_adma Pacific Digital Corporation ADMA low-level driver: 1.0
 kernel module pm80xx PMC-Sierra PM8001/8006/8081/8088/8089/8074/8076/8077/8070/8072 SAS/SATA controller driver: 0.1.38
 kernel module pmcraid PMC Sierra MaxRAID Controller Driver: 1.0.3
 kernel module qed QLogic FastLinQ 4xxxx Core Module: 8.33.0.20
 kernel module qede QLogic FastLinQ 4xxxx Ethernet Driver: 8.33.0.20
 kernel module qedf QLogic QEDF 25/40/50/100Gb FCoE Driver: 8.33.16.20
 kernel module qedi QLogic FastLinQ 4xxxx iSCSI Module: 8.33.0.20
 kernel module qla1280 Qlogic ISP SCSI (qla1x80/qla1x160) driver: 3.27.1
 kernel module qla2xxx QLogic Fibre Channel HBA Driver: 10.00.00.08-k
 kernel module qla3xxx QLogic ISP3XXX Network Driver v2.03.00-k5 : v2.03.00-k5
 kernel module qla4xxx QLogic iSCSI HBA Driver: 5.04.00-k6
 kernel module qlcnic QLogic 1/10 GbE Converged/Intelligent Ethernet Driver: 5.3.66
 kernel module qlge QLogic 10 Gigabit PCI-E Ethernet Driver : 1.00.00.35
 kernel module r6040 RDC R6040 NAPI PCI FastEthernet driver: 0.29 04Jul2016
 kernel module rsxx IBM Flash Adapter 900GB Full Height Device Driver: 4.0.3.2516
 kernel module s2io : 2.0.26.28
 kernel module sata_dwc_460ex DesignWare Cores SATA controller low level driver: 1.3
 kernel module sata_mv SCSI low-level driver for Marvell SATA controllers: 1.28
 kernel module sata_nv low-level driver for NVIDIA nForce SATA controller: 3.5
 kernel module sata_promise Promise ATA TX2/TX4/TX4000 low-level driver: 2.12
 kernel module sata_qstor Pacific Digital Corporation QStor SATA low-level driver: 0.09
 kernel module sata_sil low-level driver for Silicon Image SATA controller: 2.4
 kernel module sata_sis low-level driver for Silicon Integrated Systems SATA controller: 1.0
 kernel module sata_svw low-level driver for K2 SATA controller: 2.3
 kernel module sata_sx4 Promise SATA low-level driver: 0.12
 kernel module sata_uli low-level driver for ULi Electronics SATA controller: 1.3
 kernel module sata_via SCSI low-level driver for VIA SATA controllers: 2.6
 kernel module sata_vsc low-level driver for Vitesse VSC7174 SATA controller: 2.3
 kernel module sfc Solarflare network driver: 4.1
 kernel module sfc_falcon Solarflare Falcon network driver: 4.1
 kernel module sg SCSI generic (sg) driver: 3.5.36
 kernel module sis190 SiS sis190/191 Gigabit Ethernet driver: 1.4
 kernel module skge SysKonnect Gigabit Ethernet driver: 1.14
 kernel module sky2 Marvell Yukon 2 Gigabit Ethernet driver: 1.30
 kernel module slicoss Alacritech non-accelerated SLIC driver: 1.0
 kernel module smartpqi Driver for Microsemi Smart Family Controller version 1.1.4-130: 1.1.4-130
 kernel module smsc911x : 2008-10-21
 kernel module smsc9420 : 1.01
 kernel module snic Cisco SCSI NIC Driver: 0.0.1.18
 kernel module starfire Adaptec Starfire Ethernet driver: 2.1
 kernel module stex Promise Technology SuperTrak EX Controllers: 6.02.0000.01
 kernel module sunhme Sun HappyMealEthernet(HME) 10/100baseT ethernet driver: 3.10
 kernel module sym53c8xx NCR, Symbios and LSI 8xx and 1010 PCI SCSI adapters: 2.2.3
 kernel module tg3 Broadcom Tigon3 ethernet driver: 3.137
 kernel module thunder_bgx Cavium Thunder BGX/MAC Driver: 1.0
 kernel module thunder_xcv Cavium Thunder RGX/XCV Driver: 1.0
 kernel module tpm TPM Driver: 2.0
 kernel module tpm_atmel TPM Driver: 2.0
 kernel module tpm_crb TPM2 Driver: 0.1
 kernel module tpm_i2c_infineon TPM TIS I2C Infineon Driver: 2.2.0
 kernel module tpm_infineon Driver for Infineon TPM SLD 9630 TT 1.1 / SLB 9635 TT 1.2: 1.9.2
 kernel module tpm_nsc TPM Driver: 2.0

kernel module tpm_st33zp24 ST33ZP24 TPM 1.2 driver: 1.3.0
kernel module tpm_st33zp24_i2c STM TPM 1.2 I2C ST33 Driver: 1.3.0
kernel module tpm_tis TPM Driver: 2.0
kernel module tpm_tis_core TPM Driver: 2.0
kernel module tpm_vtpm_proxy vTPM Driver: 0.1
kernel module tulip Digital 21*4* Tulip ethernet driver: 1.1.15
kernel module typhoon 3Com Typhoon Family (3C990, 3CR990, and variants): 1.0
kernel module ufshcd_core Generic UFS host controller driver Core: 0.2
kernel module virtio_pci virtio-pci: 1
kernel module vmw_pvscsi VMware PVSCSI driver: 1.0.7.0-k
kernel module vmxnet3 VMware vmxnet3 virtual NIC driver: 1.4.16.0-k
kernel module vxlan Driver for VXLAN encapsulated traffic: 0.1
kernel module winbond_840 Winbond W89c840 Ethernet driver: 1.01-e
adduser version: 3.118
apt version: 1.8.2
apt-utils version: 1.8.2
base-files version: 10.3+deb10u4
base-passwd version: 3.5.46
bash version: 5.0-4
bsdmainutils version: 11.1.2+b1
bsdutils version: 1:2.33.1-0.1
busybox version: 1:1.30.1-4
bzip2 version: 1.0.6-9.2~deb10u1
ca-certificates version: 20190110
coreutils version: 8.30-3
cpio version: 2.12+dfsg-9
cron version: 3.0pl1-134+deb10u1
cryptsetup-bin version: 2:2.1.0-5+deb10u2
curl version: 7.64.0-4+deb10u1
dash version: 0.5.10.2-5
dbus version: 1.12.16-1
debconf version: 1.5.71
debconf-i18n version: 1.5.71
debian-archive-keyring version: 2019.1
debiantools version: 4.8.6.1
diffutils version: 1:3.7-3
dirmngr version: 2.2.12-1+deb10u1
dmidecode version: 3.2-1
dmsetup version: 2:1.02.155-3
dosfstools version: 4.1-2
dpkg version: 1.19.7
e2fsprogs version: 1.44.5-1+deb10u3
ethtool version: 1:4.19-1
fdisk version: 2.33.1-0.1
file version: 1:5.35-4+deb10u1
findutils version: 4.6.0+git+20190209-2
gcc-8-base version: 8.3.0-6
gdbm-l10n version: 1.18.1-4
gdisk version: 1.0.3-1.1
gnupg version: 2.2.12-1+deb10u1
gnupg-l10n version: 2.2.12-1+deb10u1
gnupg-utils version: 2.2.12-1+deb10u1
gpg version: 2.2.12-1+deb10u1
gpg-agent version: 2.2.12-1+deb10u1

gpg-wks-client version: 2.2.12-1+deb10u1
gpg-wks-server version: 2.2.12-1+deb10u1
gpgconf version: 2.2.12-1+deb10u1
gpgsm version: 2.2.12-1+deb10u1
gpgv version: 2.2.12-1+deb10u1
grep version: 3.3-1
groff-base version: 1.22.4-3
guile-2.2-libs version: 2.2.4+1-2+deb10u1
gzip version: 1.9-3
hdparm version: 9.58+ds-1
hostname version: 3.21
hwnfo version: 21.63-3
ifenslave version: 2.9
ifenslave-2.6 version: 2.9
ifupdown version: 0.8.35
init version: 1.56+nmu1
init-system-helpers version: 1.56+nmu1
initramfs-tools version: 0.133+deb10u1
initramfs-tools-core version: 0.133+deb10u1
iproute2 version: 4.20.0-2
iptables version: 1.8.2-4
iputils-ping version: 3:20180629-2+deb10u1
irqbalance version: 1.5.0-3
isc-dhcp-client version: 4.4.1-2
isc-dhcp-common version: 4.4.1-2
klibc-utils version: 2.0.6-1
kmod version: 26-1
kpartx version: 0.7.9-3
krb5-locales version: 1.17-3
less version: 487-0.1+b1
libacl1 version: 2.2.53-4
libaio1 version: 0.3.112-3
libapparmor1 version: 2.13.2-10
libapt-inst2.0 version: 1.8.2
libapt-pkg5.0 version: 1.8.2
libargon2-1 version: 0~20171227-0.2
libassuan0 version: 2.5.2-1
libattr1 version: 1:2.4.48-4
libaudit-common version: 1:2.8.4-3
libaudit1 version: 1:2.8.4-3
libblkid1 version: 2.33.1-0.1
libboost-atomic1.67.0 version: 1.67.0-13+deb10u1
libboost-python1.67.0 version: 1.67.0-13+deb10u1
libboost-system1.67.0 version: 1.67.0-13+deb10u1
libboost-thread1.67.0 version: 1.67.0-13+deb10u1
libbsd0 version: 0.9.1-2
libbz2-1.0 version: 1.0.6-9.2~deb10u1
libc-bin version: 2.28-10
libc6 version: 2.28-10
libcap-ng0 version: 0.7.9-2
libcap2 version: 1:2.25-2
libcap2-bin version: 1:2.25-2
libcom-err2 version: 1.44.5-1+deb10u3
libcryptsetup12 version: 2:2.1.0-5+deb10u2

libcurl4 version: 7.64.0-4+deb10u1
libdb5.3 version: 5.3.28+dfsg1-0.5
libdbus-1-3 version: 1.12.16-1
libdebconfclient0 version: 0.249
libdevmapper1.02.1 version: 2:1.02.155-3
libdns-export1104 version: 1:9.11.5.P4+dfsg-5.1
libedit2 version: 3.1-20181209-1
libelf1 version: 0.176-1.1
libestr0 version: 0.1.10-2.1
libevent-core-2.1-6 version: 2.1.8-stable-4
libevent-pthreads-2.1-6 version: 2.1.8-stable-4
libexpat1 version: 2.2.6-2+deb10u1
libext2fs2 version: 1.44.5-1+deb10u3
libfastjson4 version: 0.99.8-2
libfdisk1 version: 2.33.1-0.1
libffi6 version: 3.2.1-9
libfribidi0 version: 1.0.5-3.1+deb10u1
libgc1c2 version: 1:7.6.4-0.4
libgcc1 version: 1:8.3.0-6
libgcrypt20 version: 1.8.4-5
libgdbm-compat4 version: 1.18.1-4
libgdbm6 version: 1.18.1-4
libglib2.0-0 version: 2.58.3-2+deb10u2
libglib2.0-data version: 2.58.3-2+deb10u2
libgmp10 version: 2:6.1.2+dfsg-4
libgnutls-openssl27 version: 3.6.7-4+deb10u3
libgnutls30 version: 3.6.7-4+deb10u3
libgpg-error0 version: 1.35-1
libgsasl7 version: 1.8.0-8+b2
libgssapi-krb5-2 version: 1.17-3
libhd21 version: 21.63-3
libhogweed4 version: 3.4.1-1
libicu63 version: 63.1-6+deb10u1
libidn11 version: 1.33-2.2
libidn2-0 version: 2.0.5-1+deb10u1
libip4tc0 version: 1.8.2-4
libip6tc0 version: 1.8.2-4
libiptc0 version: 1.8.2-4
libisc-export1100 version: 1:9.11.5.P4+dfsg-5.1
libjson-c3 version: 0.12.1+ds-2
libk5crypto3 version: 1.17-3
libkeyutils1 version: 1.6-6
libklibc version: 2.0.6-1
libkmod2 version: 26-1
libkrb5-3 version: 1.17-3
libkrb5support0 version: 1.17-3
libksba8 version: 1.3.5-2
libkyotocabinet16v5 version: 1.2.76-4.2+b1
libldap-2.4-2 version: 2.4.47+dfsg-3+deb10u2
libldap-common version: 2.4.47+dfsg-3+deb10u2
liblocale-gettext-perl version: 1.07-3+b4
liblognorm5 version: 2.0.5-1
libltdl7 version: 2.4.6-9
liblz4-1 version: 1.8.3-1

liblzma5 version: 5.2.4-1
liblzo2-2 version: 2.10-0.1
libmagic-mgc version: 1:5.35-4+deb10u1
libmagic1 version: 1:5.35-4+deb10u1
libmailutils5 version: 1:3.5-3
libmariadb3 version: 1:10.3.22-0+deb10u1
libmnl0 version: 1.0.4-2
libmount1 version: 2.33.1-0.1
libmpdec2 version: 2.4.2-2
libncurses6 version: 6.1+20181013-2+deb10u2
libncursesw6 version: 6.1+20181013-2+deb10u2
libnetfilter-contrack3 version: 1.0.7-1
libnettle6 version: 3.4.1-1
libnewt0.52 version: 0.52.20-8
libnfnetwork0 version: 1.0.1-3+b1
libnftnl11 version: 1.1.2-2
libnghttp2-14 version: 1.36.0-2+deb10u1
libnpt0 version: 1.6-1
libntlm0 version: 1.5-1
libnuma1 version: 2.0.12-1
libopenipmi0 version: 2.0.25-2.1
libopts25 version: 1:5.18.12-4
libp11-kit0 version: 0.23.15-2
libpam-modules version: 1.3.1-5
libpam-modules-bin version: 1.3.1-5
libpam-runtime version: 1.3.1-5
libpam-systemd version: 241-7~deb10u4
libpam0g version: 1.3.1-5
libpcap0.8 version: 1.8.1-6
libpci3 version: 1:3.5.2-1
libpcre3 version: 2:8.39-12
libperl5.28 version: 5.28.1-6
libpopt0 version: 1.16-12
libprocps7 version: 2:3.3.15-2
libpsl5 version: 0.20.2-2
libpython2.7 version: 2.7.16-2+deb10u1
libpython2.7-minimal version: 2.7.16-2+deb10u1
libpython2.7-stdlib version: 2.7.16-2+deb10u1
libpython3.7 version: 3.7.3-2+deb10u1
libpython3.7-minimal version: 3.7.3-2+deb10u1
libpython3.7-stdlib version: 3.7.3-2+deb10u1
libreadline7 version: 7.0-5
librtmp1 version: 2.4+20151223.gitfa8646d.1-2
libsasl2-2 version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules-db version: 2.1.27+dfsg-1+deb10u1
libseccomp2 version: 2.3.3-4
libselinux1 version: 2.8-1+b1
libsemanage-common version: 2.8-2
libsemanage1 version: 2.8-2
libsensors-config version: 1:3.5.0-3
libsensors5 version: 1:3.5.0-3
libsepol1 version: 2.8-1
libsgutils2-2 version: 1.44-1

libslang2 version: 2.3.2-2
libsmartcols1 version: 2.33.1-0.1
libsnmp-base version: 5.7.3+dfsg-5
libsnmp30 version: 5.7.3+dfsg-5
libsqlite3-0 version: 3.27.2-3
libss2 version: 1.44.5-1+deb10u3
libssh2-1 version: 1.8.0-2.1
libssl1.1 version: 1.1.1d-0+deb10u3
libstdc++6 version: 8.3.0-6
libsysfs2 version: 2.1.0+repack-5
libsystemd0 version: 241-7~deb10u4
libtasn1-6 version: 4.13-3
libtext-charwidth-perl version: 0.04-7.1+b1
libtext-iconv-perl version: 1.7-5+b7
libtext-wrapi18n-perl version: 0.06-7.1
libtinfo6 version: 6.1+20181013-2+deb10u2
libuchardet0 version: 0.0.6-3
libudev1 version: 241-7~deb10u4
libunistring2 version: 0.9.10-1
liburcu6 version: 0.10.2-1
libuuid1 version: 2.33.1-0.1
libwrap0 version: 7.6.q-28
libx11-6 version: 2:1.6.7-1
libx11-data version: 2:1.6.7-1
libx86emu2 version: 2.0-1
libxau6 version: 1:1.0.8-1+b2
libxcb1 version: 1.13.1-2
libxdmcp6 version: 1:1.1.2-3
libxext6 version: 2:1.3.3-1+b2
libxml2 version: 2.9.4+dfsg1-7+b3
libxmuu1 version: 2:1.1.2-2+b3
libxtables12 version: 1.8.2-4
libyajl2 version: 2.1.0-3
libzstd1 version: 1.3.8+dfsg-3
linux-base version: 4.6
linux-firmware version: 1.183.2
login version: 1:4.5-1.1
logrotate version: 3.14.0-4
lsb-base version: 10.2019051400
lsscsi version: 0.30-0.1
mailutils version: 1:3.5-3
mailutils-common version: 1:3.5-3
mariadb-common version: 1:10.3.22-0+deb10u1
mawk version: 1.3.3-17+b3
mime-support version: 3.62
mount version: 2.33.1-0.1
multipath-tools version: 0.7.9-3
multipath-tools-boot version: 0.7.9-3
mysql-common version: 5.8+1.0.5
nano version: 3.2-3
ncurses-base version: 6.1+20181013-2+deb10u2
ncurses-bin version: 6.1+20181013-2+deb10u2
ncurses-term version: 6.1+20181013-2+deb10u2
net-tools version: 1.60+git20180626.aebd88e-1

netbase version: 5.6
nload version: 0.7.4-2+b1
ntp version: 1:4.2.8p12+dfsg-4
numactl version: 2.0.12-1
openipmi version: 2.0.25-2.1
openssh-client version: 1:7.9p1-10+deb10u2
openssh-server version: 1:7.9p1-10+deb10u2
openssh-sftp-server version: 1:7.9p1-10+deb10u2
openssl version: 1.1.1d-0+deb10u3
passwd version: 1:4.5-1.1
perl version: 5.28.1-6
perl-base version: 5.28.1-6
perl-modules-5.28 version: 5.28.1-6
pigz version: 2.4-1
pinentry-curses version: 1.1.0-2
powermgmt-base version: 1.34
procps version: 2:3.3.15-2
publicsuffix version: 20190415.1030-1
python3.7 version: 3.7.3-2+deb10u1
python3.7-minimal version: 3.7.3-2+deb10u1
qemu-guest-agent version: 1:3.1+dfsg-8+deb10u5
readline-common version: 7.0-5
rsyslog version: 8.1901.0-1
runit-helper version: 2.8.6
sdparm version: 1.10-1
sed version: 4.7-1
sensible-utils version: 0.0.12
sg3-utils version: 1.44-1
sg3-utils-udev version: 1.44-1
shared-mime-info version: 1.10-1
smartmontools version: 6.6-1
smp-utils version: 0.98-2
snmp version: 5.7.3+dfsg-5
snmpd version: 5.7.3+dfsg-5
snmp version: 1:4.2.8p12+dfsg-4
ssmtp version: 2.64-8.1
sysstat version: 12.0.3-2
systemd version: 241-7~deb10u4
systemd-sysv version: 241-7~deb10u4
sysvinit-utils version: 2.93-8
tar version: 1.30+dfsg-6
tasksel version: 3.53
tasksel-data version: 3.53
tcpdump version: 4.9.3-1~deb10u1
tofrodos version: 1.7.13+ds-4
traceroute version: 1:2.1.0-2
tzdata version: 2020a-0+deb10u1
ucf version: 3.0038+nmu1
udev version: 241-7~deb10u4
util-linux version: 2.33.1-0.1
vim-common version: 2:8.1.0875-5
vim-tiny version: 2:8.1.0875-5
whiptail version: 0.52.20-8
xauth version: 1:1.0.10-1

xdg-user-dirs version: 0.17-2
xxd version: 2:8.1.0875-5
xz-utils version: 5.2.4-1
zlib1g version: 1:1.2.11.dfsg-1
linux-firmware: 1.183.2
ssmtp: 2.64-8.1

Elasticsearch-specific and additional Caringo distributions:

Elasticsearch 6.8.6 / 5.6.12
elasticsearch-curator 4.3.1 (supports Elasticsearch 5 and 6)
txes 0.1.4+
Swarm S3 Backup Restore 1.2.1
Swarm Search 6.3.1
Swarm Metrics 6.3.1

Swarm Storage 11.2 Release

- [New Features](#)
- [Additional Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Known Issues](#)
- [Upgrading Swarm](#)

New Features

Expanded SEND – SCSP SEND, an admin-only method that allows forcing an object to be written immediately in another cluster, now works with every type of and any number of Swarm feeds: replication, search, and S3 backup. The expanded functionality works through several new query arguments, two to specify which feed IDs or types to target and one to control the timeout (if any) for the request to complete. See [SCSP SEND](#). (SWAR-8441)

Elasticsearch – This release focuses on changes that make it easier to monitor and manage Elasticsearch, Swarm Search, and Swarm Metrics:

- The Swarm Search RPM installation now checks and warns if `firewalld` is enabled, reminding to check the firewall rules for ports 9200 and 9300, which are needed by Elasticsearch. (SWAR-8416)
- Swarm dynamically updates DNS lookups after Elasticsearch nodes are restarted. (SWAR-8817)
- The Swarm Metrics curator is now independent of `HTTP_PROXY` and related shell environment variables and so is less subject to disruption. (SWAR-8452)
- The Swarm Metrics curator has improved defaults for its logging, increased to 10 logs and up to 10 MB. (SWAR-8401)

This release also includes changes to help with Swarm management and performance:

- Swarm now ships with the Prometheus Node Exporter enabled and configured to work by default, to simplify implementation and avoid rebooting. To disable the Node Exporter on a node, set `"metrics.enabledNodeExporter=False"` in the node's configuration file; to disable across the entire cluster, set `metrics.nodeExporterFrequency` to 0. (SWAR-8578)
- Swarm's inter-process locking process has been reworked, granting a small performance gain for larger clusters and a reduction in related WARNING-level log messages. (SWAR-8835)
- Swarm has restored performance for clients who have not migrated from legacy authentication/authorization. (SWAR-8810)

Additional Changes

These items are other changes, including those that come from testing and user feedback.

- **OSS Versions** – See [Third-Party Components for 11.2](#) for the complete listing of packages and versions.
- **Fixed in 11.2.0**
 - The multipart write 202 response now includes Location headers of the resulting manifests that are analogous to the Location headers of a normal EC write. (SWAR-8886)
 - Resolved an error in the assessment of licensed space usage that prevented a node from accepting writes. (SWAR-8869)
 - Resolved an issue related to TCP window sizes that can cause socket disconnects, pauses, and hangs. (SWAR-8847)
 - Resolved an issue that can lead to a node crash in large clusters. (SWAR-8832)
 - Basic auth of the admin user for special administrative SCSP requests did not correctly handle a stored hashed admin password. (SWAR-8814)
 - Infrequent WARNING messages, "Node/Volume entry not published due to lock contention (...); action will be retried," may appear in logs. (SWAR-8802)

- Resolved an issue causing rebooted Swarm nodes to allow client requests before mounting all volumes. (SWAR-8801)

Upgrade Impacts

Use the supported versions of Swarm components for the target version of Elasticsearch:

Elasticsearch 6.8.6	Swarm Storage 11.1 - 11.2	Gateway 6.3	SwarmFS 2.4	Recommended configuration.
Elasticsearch 5.6.12	Swarm Storage 10.0 - 11.2	Gateway 6.0 - 6.3	SwarmFS 2.4	Plan to migrate to Elasticsearch 6 . Support for earlier versions is ending.
Elasticsearch 2.3.3	Swarm Storage 9.6 - 11.2	Gateway 5.4	SwarmFS 2.1	

These items are changes to the product function that may require operational or development changes for integrated applications. Address the upgrade impacts for each of the versions since the currently running version:

Impacts for 11.2

- Upgrading Elasticsearch** – Elasticsearch 5.6.12/2.3.3 may be used with Storage 11 if move to ES 6 cannot be performed immediately, but start the migration now (see [Migrating from Older Elasticsearch](#)). Support for ES 5.6.12/2.3.3 ends in a future release, and testing for 2.3.3 with Swarm 11 is discontinued. *Important:* Always upgrade Swarm Search and Metrics at the same time upgrading ES. Do not run an ES 5 Search or Metrics Curator against ES 6.
- Rolling upgrade** – During a rolling upgrade, the mixed state in Swarm versions among nodes may cause errors in the Swarm UI (and in management API calls). Use the legacy Admin Console (port 90) to monitor the rolling upgrade. (SWAR-8716)
- Settings changes** – These settings are new with this release:
 - `scsp.defaultFeedSendTimeout`, (default 30 seconds) a non-persisted node-level setting that sets the timeout on a feed SEND request, if the `timeout=true` query argument is provided. (SWAR-8441).
 - `chassis.name`, (default blank), a node-level setting that stores a user-defined chassis name. (SWAR-8823)
- Differences in `scsp.forceLegacyNonce` configuration** depending on the version upgrading from (SWAR-9020):
- If currently running a Swarm Storage version prior to 11.1**, and upgrading to 11.1, 11.2, 11.3, 12.0 or 12.1:

Before upgrading, set `scsp.forceLegacyNonce=true` in the `node.cfg` file. After the upgrade, when the cluster is fully up, update `scsp.forceLegacyNonce=false` using `swarmctl` and change `scsp.forceLegacyNonce=false` in the `node.cfg` file.

If currently running a Swarm Storage version 11.1, 11.2, 11.3, 12.0 or 12.1 and upgrading to another version from that list:

Before upgrading, verify `scsp.forceLegacyNonce=false` is in the `node.cfg` file and verify using `swarmctl` that `scsp.forceLegacyNonce=false` in the cluster.

Use `swarmctl` to check or change settings

Use `'swarmctl -C scsp.forceLegacyNonce'` to check the value of `scsp.forceLegacyNonce`.

Use `'swarmctl -C scsp.forceLegacyNonce -V False'` to set the value to false.

For more details, see <https://support.cloud.caringo.com/tools/Tech-Support-Scripts-Bundle-swarmctl.pdf>.

Impacts for 11.1

- **Upgrading Elasticsearch** – Use Elasticsearch 5.6.12/2.3.3 with Storage 11.1 if moving to ES 6 immediately is not possible, but start the migration now (see [Migrating from Older Elasticsearch](#)). Support for ES 5.6.12/2.3.3 ends in a future release, and testing for 2.3.3 with Swarm 11 is discontinued. *Important:* Always upgrade Swarm Search and Metrics at the same time ES is upgraded. Do not run an ES 5 Search or Metrics Curator against ES 6.
- **Swarm Search and Metrics** – This release includes new versions of Swarm Search and Metrics RPMs. Both require Python 3 to be installed on the ES servers they run on.
 - For Swarm Metrics on RHEL/CentOS 7.7, first install this dependency: `yum install epel-release`
- **Python 3** – Install Python 3 if is not automatically installed with RHEL/CentOS 7.
- **Propagate Delete Removed** – For [replication and S3 backup feeds](#), the Propagate Deletes option is removed from the legacy Admin Console and the Management API (propagateDeletes, nodeDeletes fields). (SWAR-8609, SWAR-8615)
- **Swarm Configuration** – Run the [Storage Settings Checker](#) before upgrading to this version, to identify configuration issues.
 - The Storage Settings Checker now requires Python 3 to be installed. (SWAR-8742)
 - `crier.deadVolumeWall` has been unpublished for reimplement. (SWAR-8640)
- **S3 Backup Restore** – The S3 Backup Restore Tool has been migrated to Python 3.6. If the tool is installed, uninstall it and [install the new version](#). (SWAR-8703)
- **Upgrade Process** – During the upgrade to 11.1, it may not be possible to monitor the cluster via the Swarm UI. Workaround: Use the legacy Admin Console (port 90) during upgrade. (SWAR-8716)
- **Differences in `scsp.forceLegacyNonce` configuration** depending on the version being upgraded from (SWAR-9020):
- **If currently running a Swarm Storage version prior to 11.1**, and upgrading to 11.1, 11.2, 11.3, 12.0 or 12.1:

Before upgrading, set `scsp.forceLegacyNonce=true` in the `node.cfg` file. After the upgrade, when the cluster is fully up, update `scsp.forceLegacyNonce=false` using `swarmctl` and change `scsp.forceLegacyNonce=false` in the `node.cfg` file.

If currently running a Swarm Storage version 11.1, 11.2, 11.3, 12.0 or 12.1 and upgrading to another version from that list:

Before upgrading, verify `scsp.forceLegacyNonce=false` is in the `node.cfg` file and verify using `swarmctl` that `scsp.forceLegacyNonce=false` in the cluster.

Use `swarmctl` to check or change settings

Use `'swarmctl -C scsp.forceLegacyNonce'` to check the value of `scsp.forceLegacyNonce`.

Use `'swarmctl -C scsp.forceLegacyNonce -V False'` to set the value to `false`.

For more details, see <https://support.cloud.caringo.com/tools/Tech-Support-Scripts-Bundle-swarmctl.pdf>.

Impacts for 11.0

- **Upgrading Elasticsearch** – You may use Elasticsearch 2.3.3 with Storage 11.0 if you cannot move to 5.6 now, but plan the migration immediately (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release, and testing with Swarm 11 is discontinuing.
- **Propagate Deletes Deprecated** – The option to disable *Propagate Deletes* on [replication feeds](#) is deprecated; use [Object Versioning](#) to preserve deleted content. Do not disable Propagate Deletes when versioning is enabled or when defining an S3 Backup. (SWAR-8609)
- **Configuration Settings** – Run the [Storage Settings Checker](#) to identify these and other configuration issues.
 - Changed settings:

- `ec.segmentConsolidationFrequency` (`ecSegmentConsolidationFrequency` in SNMP) has an improved default (10), which you must apply to your cluster when you upgrade. (SWAR-8483)
- `cluster.name` is now required. Add it to the `cluster.cfg` file. (SWAR-8466).
- `metrics.nodeExporterFrequency` (`metricsExporterFrequency` in SNMP) is now a persisted cluster setting. (SWAR-8467).
- Removed settings:
 - `chassis.processes` is allowed but is ignored.
- Numerous settings are now promoted to *cluster-level* (versus node-level) scope, so you can manage them via **Settings > Cluster** in the Swarm UI (SWAR-8457):
 - `console.expiryErrInterval`
 - `console.expiryWarnInterval`
 - `console.indexErrorLevel`
 - `console.indexWarningLevel`
 - `console.port`
 - `console.reportStyleUrl`
 - `console.spaceErrorLevel`
 - `console.spaceWarnLevel`
 - `console.styleUrl`
 - `feeds.retry`
 - `feeds.statsReportInterval`
 - `health.parallelWriteTimeout`
 - `log.obscureUUIDs`
 - `metrics.enableNodeExporter`
 - `network.dnsDomain`
 - `network.dnsServers`
 - `network.icmpAcceptRedirects`
 - `network.igmpVersion`
 - `network.mtu`
 - `startup.certificates`

Impacts for 10.2

- **Upgrading Elasticsearch** – You may continue to use Elasticsearch 2.3.3 with Storage 10.2 until you are able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release. the upgrade to Elasticsearch 5.6 must be completed before [upgrading to Gateway 6.0](#).
- **Configuration Settings** – Run the [Storage Settings Checker](#) before any Swarm 10 upgrade to identify configuration issues. Note these changes:
 - `ec.protectionLevel` is now persisted. (SWAR-8231)
 - `index.ovMinNodes=3` is the new default for the overlay index, in support of Swarm 10's new architecture. To keep your overlay index operational, set this new value in your cluster, through the UI or by SNMP (`overlayMinNodes`). (SWAR-8278)
 - `metrics.enableNodeExporter` can be set to True, which enables the Prometheus Node Exporter on that node. (SWAR-8408, SWAR-8578)
 - `metrics.nodeExporterFrequency`, a new dynamic setting, sets how frequently to refresh Swarm-specific Prometheus metrics in Elasticsearch; it defaults to 0, which disables this export. (SWAR-8408).

Impacts for 10.1

- **Upgrading Elasticsearch** – Continue to use Elasticsearch 2.3.3 with Storage 10.1 until able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release. Complete the upgrade to Elasticsearch 5.6 before upgrading to Gateway 6.0.
- **Configuration Settings** – Run the [Storage Settings Checker](#) before any Swarm 10 upgrade to identify configuration issues.
 - `metrics.enableNodeExporter=true` enables Swarm to run the Prometheus node exporter on port 9100. (SWAR-8170)
- **IP address update delay** – When upgrading from Swarm 9 to the new architecture of Swarm 10, note the "ghosts" of previously used IP addresses may appear in the Storage UI; these resolve within 4 days. (SWAR-8351)
- **Update MIBs on CSN** – Before upgrading to Storage 10.x, the MIBs on the CSN must be updated. From the Swarm Support tools bundle, run the `platform-update-mibs.sh` script. (CSN-1872)

Impacts for 10.0

- **Upgrading Elasticsearch** – You may continue to use Elasticsearch 2.3.3 with Storage 10.0 until you are able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release.
- **Configuration Settings** – Run the [Storage Settings Checker](#) to identify these and other configuration issues.
 - Changes for the new single-IP dense architecture:
 - `network.ipAddress` – multiple IP addresses now disallowed
 - `chassis.processes` – removed; multi-server configurations are no longer supported
 - `ec.protectionLevel` – new value "volume"
 - `ec.subclusterLossTolerance` – removed
 - Changes for security (see next section)
 - `security.administrators`, `security.operators` – removed 'snmp' user
 - `snmp.rwCommunity`, `snmp.roCommunity` – new settings for 'snmp' user
 - `startup.certificates` – new setting to hold any and all public keys
 - New settings:
 - `disk.atimeEnabled`
 - `health.parallelWriteTimeout`
 - `search.pathDelimiter`
- **Required SNMP security change** – Remove the `snmp` key from the `security.administrators` setting, and update `snmp.rwCommunity` with its value. Nodes that contain only the `snmp` key in the `security.administrators` setting does not boot. If you changed the default value of the `snmp` key in the `security.operators` setting, update `snmp.roCommunity` with that value and then remove the `snmp` key from `security.operators`. In the `security.operators` setting, 'snmp' is a reserved key, and it cannot be an authorized console operator name. (SWAR-8097)
- **EC protection**
 - *Best practice:* Use `ec.protectionLevel=node`, which distributes segments across the cluster's physical/virtual machines. Do not use `ec.protectionLevel=subcluster` unless you already have subclusters defined and are sure the specified EC encoding is supported. A new level, `ec.protectionLevel=volume`, allows EC writes to succeed if you have a small cluster with fewer than $(k+p)/p$ nodes. (Swarm always seeks the highest protection possible for EC segments, regardless of the level you set.)
 - Optimize hardware for EC by verifying there are more than $k+p$ subclusters/nodes (as set by `ec.protectionLevel`); for example, with `policy.ecEncoding=5:2`, you need at least 8 subclusters/nodes. When Swarm cannot distribute EC segments adequately for protection, EC writes can fail despite ample free space. (SWAR-7985)
 - Setting `ec.protectionLevel=subcluster` without creating subclusters (defining `node.subcluster` across sets of nodes) causes a critical error and lowers the protection level to 'node'. (SWAR-8175)

- **Small clusters** – Verify the following settings if using 10 or fewer Swarm nodes. Do not use fewer than 3 in production. *Important:* If you need to change any, do so *before* upgrading to Swarm 10.
 - **policy.replicas** – The `min` and `default` values for numbers of replicas to keep in your cluster must not exceed your number of nodes. For example, a 3-node cluster may have only `min=2` or `min=3`.
 - **EC encoding and protection** – For EC encoding, verify you have enough nodes to support the cluster's encoding (`policy.ecEncoding`). For EC writes to succeed with fewer than $(k+p)/p$ nodes, use the new level, `ec.protectionLevel=volume`.
 - *Best practice:* Keep at least one physical machine in your cluster beyond the minimum number needed. This allows for one machine to be down for maintenance without compromising the constraint.
- **"Cluster in a box"** – Swarm supports a "cluster in a box" configuration as long as that box is running a virtual machine host and Swarm instances are running in 3 or more VMs. Each VM boots separately and has its own IP address. Follow the recommendations for small clusters, substituting VMs for nodes. If you have two physical machines, use the "cluster in a box" configuration, but move to direct booting of Swarm with 3 or more.
- **Offline node status** – Because Swarm 10's new architecture reduces the number of IP addresses in your storage cluster, you may see the old IPs and subclusters reporting as **Offline** nodes until they timeout in 4 days (`crier.forgetOfflineInterval`), which is expected.

Watch Items and Known Issues

The following operational limitations and watch items exist in this release.

- If a node mounts an encrypted volume that is missing the encryption key in the configuration, the node fails to mount all disks in the node. (SWAR-8762)
- S3 Backup feeds do not backup logical objects greater than 5 GB. (SWAR-8554)
- If downgrading from Swarm 11.0, CRITICAL errors may appear on the feeds. To stop the errors, edit the existing feed definition names via the Swarm UI or legacy Admin Console. (SWAR-8543)
- When restarting a cluster of virtual machines that are UEFI-booted (versus legacy BIOS), the chassis shut down but do not come back up. (SWAR-8054)
- If wiping the Elasticsearch cluster, the Storage UI shows no NFS config. Contact DataCore Support for help repopulating the SwarmFS config information. (SWAR-8007)
- If bucket is deleted, any incomplete multipart upload into that bucket leaves the parts (unnamed streams) in the domain. To find and delete them, use the `s3cmd` utility (search the Support site for "s3cmd" for guidance). (SWAR-7690)
- Logs showed the error "FEEDS WARNING: calcFeedInfo(etag=xxx) cannot find domain xxx, which is needed for a domains-specific replication feed". The root cause is fixed; if such warnings are received, contact DataCore Support so the issue can be resolved. (SWAR-7556)
- With multipath-enabled hardware, the Swarm console Disk Volume Menu may erroneously show too many disks, having multiplied the actual disks in use by the number of possible paths to them. (SWAR-7248)

Note these installation issues:

- The `elasticsearch-curator` package may show an error during an upgrade, which is a known curator issue. Workaround: Reinstall the curator: `yum reinstall elasticsearch-curator` (SWAR-7439)
- Do not install the Swarm Search RPM before installing Java. If Gateway startup fails with "Caringo script plugin is missing from indexer nodes", uninstall and reinstall the Swarm Search RPM. (SWAR-7688)

Upgrading Swarm

Proceed to [How to Upgrade Swarm](#) to upgrade Swarm 9 or higher.



Important

Contact DataCore Support for guidance if needing to migrate from Swarm 8.x or earlier.

Third-Party Components for 11.2

zope.interface version: 4.7.1
 ipaddress version: 1.0.23
 cryptography version: 2.8
 pyOpenSSL version: 19.1.0
 service_identity version: 18.1.0
 incremental version: 17.5.0
 Twisted[tls] version: 19.10.0
 pyutil version: 3.3.0
 python-dateutil version: 2.8.1
 Werkzeug version: 0.16.0
 klein version: 19.6.0
 requests version: 2.21.0
 zfec version: 1.5.3
 yajl-py version: 2.1.2
 certifi version: *latest as of 2020-06-19*
 pyratemp version: 0.3.2
 Newt version: 0.52.20
 Prometheus node_exporter version: 0.18.1
 libpng version: 1.2.8
 LILO version: 22.7.1
 Operating system: Debian GNU/Linux 10 (buster)
 Linux kernel: 4.19.84
 kernel module 3w_9xxx 3ware 9000 Storage Controller Linux Driver: 2.26.02.014
 kernel module 3w_sas LSI 3ware SAS/SATA-RAID Linux Driver: 3.26.02.000
 kernel module 3w_xxxx 3ware Storage Controller Linux Driver: 1.26.02.003
 kernel module 8021q : 1.8
 kernel module 8139cp RealTek RTL-8139C+ series 10/100 PCI Ethernet driver: 1.3
 kernel module 8139too RealTek RTL-8139 Fast Ethernet driver: 0.9.28
 kernel module aacraid Dell PERC2, 2/Si, 3/Si, 3/Di, Adaptec Advanced Raid Products, HP NetRAID-4M, IBM ServeRAID & ICP SCSI driver: 1.2.1
 [50877]-custom
 kernel module acard_ahci ACard AHCI SATA low-level driver: 1.0
 kernel module ad7418 AD7416/17/18 driver: 0.4
 kernel module ahci AHCI SATA low-level driver: 3.0
 kernel module aic79xx Adaptec AIC790X U320 SCSI Host Bus Adapter driver: 3.0
 kernel module aic7xxx Adaptec AIC77XX/78XX SCSI Host Bus Adapter driver: 7.0
 kernel module aic94xx Adaptec aic94xx SAS/SATA driver: 1.0.3
 kernel module am53c974 AM53C974 SCSI driver: 1.00
 kernel module amd_xgbe AMD 10 Gigabit Ethernet Driver: 1.0.3
 kernel module arcmsr Areca ARC11xx/12xx/16xx/188x SAS/SATA RAID Controller Driver: v1.40.00.09-20180709
 kernel module ata_generic low-level driver for generic ATA: 0.2.15
 kernel module ata_piix SCSI low-level driver for Intel PIIX/ICH ATA controllers: 2.13
 kernel module atl1 Atheros L1 Gigabit Ethernet Driver: 2.1.3
 kernel module atl1c Qualcomm Atheros 100/1000M Ethernet Network Driver: 1.0.1.1-NAPI
 kernel module atl1e Atheros 1000M Ethernet Network Driver: 1.0.0.7-NAPI
 kernel module atl2 Atheros Fast Ethernet Network Driver: 2.2.3
 kernel module atlantic aQuantia Corporation(R) Network Driver: 2.0.3.0-kern
 kernel module atxp1 System voltages control via Attansic ATXP1: 0.6.3
 kernel module b44 Broadcom 44xx/47xx 10/100 PCI ethernet driver: 2.0
 kernel module be2iscsi Emulex OneConnectOpen-iSCSI Driver version11.4.0.1 Driver 11.4.0.1: 11.4.0.1
 kernel module be2net Emulex OneConnect NIC Driver 12.0.0.0: 12.0.0.0

kernel module bfa QLogic BR-series Fibre Channel HBA Driver fcpim: 3.2.25.1
 kernel module bna QLogic BR-series 10G PCIe Ethernet driver: 3.2.25.1
 kernel module bnx2 QLogic BCM5706/5708/5709/5716 Driver: 2.2.6
 kernel module bnx2fc QLogic FCoE Driver: 2.11.8
 kernel module bnx2i QLogic NetXtreme II BCM5706/5708/5709/57710/57711/57712/57800/57810/57840 iSCSI Driver: 2.7.10.1
 kernel module bnx2x QLogic BCM57710/57711/57711E/57712/57712_MF/57800/57800_MF/57810/57810_MF/57840/57840_MF Driver: 1.712.30-0
 kernel module bnxt_en Broadcom BCM573xx network driver: 1.9.2
 kernel module bonding Ethernet Channel Bonding Driver, v3.7.1: 3.7.1
 kernel module cnic QLogic cnic Driver: 2.5.22
 kernel module csiostor Chelsio FCoE driver: 1.0.0-ko
 kernel module cxgb3 Chelsio T3 Network Driver: 1.1.5-ko
 kernel module cxgb3i Chelsio T3 iSCSI Driver: 2.0.1-ko
 kernel module cxgb4 Chelsio T4/T5/T6 Network Driver: 2.0.0-ko
 kernel module cxgb4i Chelsio T4-T6 iSCSI Driver: 0.9.5-ko
 kernel module cxgb4vf Chelsio T4/T5/T6 Virtual Function (VF) Network Driver: 2.0.0-ko
 kernel module dca : 1.12.1
 kernel module dcdbas Dell Systems Management Base Driver (version 5.6.0-3.2): 5.6.0-3.2
 kernel module de2104x Intel/Digital 21040/1 series PCI Ethernet driver: 0.7
 kernel module dmfe Davicom DM910X fast ethernet driver: 1.36.4
 kernel module e100 Intel(R) PRO/100 Network Driver: 3.5.24-k2-NAPI
 kernel module e1000 Intel(R) PRO/1000 Network Driver: 7.3.21-k8-NAPI
 kernel module e1000e Intel(R) PRO/1000 Network Driver: 3.2.6-k
 kernel module eeprom_93cx6 EEPROM 93cx6 chip driver: 1.0
 kernel module efivars sysfs interface to EFI Variables: 0.08
 kernel module ena Elastic Network Adapter (ENA): 1.5.0K
 kernel module enic Cisco VIC Ethernet NIC Driver: 2.3.0.53
 kernel module esas2r esas2r: 1.00
 kernel module esp_scsi ESP SCSI driver core: 2.000
 kernel module fm10k Intel(R) Ethernet Switch Host Interface Driver: 0.23.4-k
 kernel module fnic Cisco FCoE HBA Driver: 1.6.0.34
 kernel module hpsa Driver for HP Smart Array Controller version 3.4.20-125: 3.4.20-125
 kernel module i40e Intel(R) 40-10 Gigabit Ethernet Connection Network Driver: 2.7.29
 kernel module i40e Intel(R) Ethernet Connection XL710 Network Driver: 2.3.2-k
 kernel module i40evf Intel(R) XL710 X710 Virtual Function Network Driver: 3.2.2-k
 kernel module ice Intel(R) Ethernet Connection E800 Series Linux Driver: ice-0.7.0-k
 kernel module igb Intel(R) Gigabit Ethernet Network Driver: 5.4.0-k
 kernel module igbvf Intel(R) Gigabit Virtual Function Network Driver: 2.4.0-k
 kernel module ioatdma : 4.00
 kernel module ipmi_msghandler Incoming and outgoing message routing for an IPMI interface.: 39.2
 kernel module ipr IBM Power RAID SCSI Adapter Driver: 2.6.4
 kernel module ips IBM ServeRAID Adapter Driver 7.12.05: 7.12.05
 kernel module iscsi : 1.2.0
 kernel module ixgb Intel(R) PRO/10GbE Network Driver: 1.0.135-k2-NAPI
 kernel module ixgbe Intel(R) 10 Gigabit PCI Express Network Driver: 5.1.0-k
 kernel module ixgbe Intel(R) 10GbE PCI Express Linux Network Driver: 5.5.5
 kernel module ixgbevfn Intel(R) 10 Gigabit Virtual Function Network Driver: 4.1.0-k
 kernel module jme JMicron JMC2x0 PCI Express Ethernet driver: 1.0.8
 kernel module libcxgb Chelsio common library: 1.0.0-ko
 kernel module libcxgbi Chelsio iSCSI driver library: 0.9.1-ko
 kernel module liquidio Cavium LiquidIO Intelligent Server Adapter Driver: 1.7.2
 kernel module liquidio_vf Cavium LiquidIO Intelligent Server Adapter Virtual Function Driver: 1.7.2
 kernel module lpfc Emulex LightPulse Fibre Channel SCSI driver 12.0.0.6: 0
 kernel module megaraid LSI Logic MegaRAID legacy driver: 2.00.4

kernel module megaraid_mbox LSI Logic MegaRAID Mailbox Driver: 2.20.5.1
kernel module megaraid_mm LSI Logic Management Module: 2.20.2.7
kernel module megaraid_sas Avago MegaRAID SAS Driver: 07.706.03.00-rc1
kernel module mlx4_core Mellanox ConnectX HCA low-level driver: 4.0-0
kernel module mlx4_en Mellanox ConnectX HCA Ethernet driver: 4.0-0
kernel module mlx5_core Mellanox 5th generation network adapters (ConnectX series) core driver: 5.0-0
kernel module mpt3sas LSI MPT Fusion SAS 3.0 Device Driver: 26.100.00.00
kernel module mptbase Fusion MPT base driver: 3.04.20
kernel module mptctl Fusion MPT misc device (ioctl) driver: 3.04.20
kernel module mptfc Fusion MPT FC Host driver: 3.04.20
kernel module mptsas Fusion MPT SAS Host driver: 3.04.20
kernel module mptscsih Fusion MPT SCSI Host driver: 3.04.20
kernel module mptspi Fusion MPT SPI Host driver: 3.04.20
kernel module mtip32xx Micron RealSSD PCIe Block Driver: 1.3.1
kernel module mvsas Marvell 88SE6440 SAS/SATA controller driver: 0.8.16
kernel module myri10ge Myricom 10G driver (10GbE): 1.5.3-1.534
kernel module netxen_nic QLogic/NetXen (1/10) GbE Intelligent Ethernet Driver: 4.0.82
kernel module nfp The Netronome Flow Processor (NFP) driver.: 4.19.84
kernel module nicpf Cavium Thunder NIC Physical Function Driver: 1.0
kernel module nicvf Cavium Thunder NIC Virtual Function Driver: 1.0
kernel module niu NIU ethernet driver: 1.1
kernel module nvme : 1.0
kernel module nvme_core : 1.0
kernel module pata_acpi SCSI low-level driver for ATA in ACPI mode: 0.2.3
kernel module pata_ali low-level driver for ALi PATA: 0.7.8
kernel module pata_amd low-level driver for AMD and Nvidia PATA IDE: 0.4.1
kernel module pata_artop SCSI low-level driver for ARTOP PATA: 0.4.6
kernel module pata_atiixp low-level driver for ATI IXP200/300/400: 0.4.6
kernel module pata_atp867x low level driver for Artop/Acard 867x ATA controller: 0.7.5
kernel module pata_cmd64x low-level driver for CMD64x series PATA controllers: 0.2.18
kernel module pata_efar SCSI low-level driver for EFAR PIIX clones: 0.4.5
kernel module pata_hpt366 low-level driver for the Highpoint HPT366/368: 0.6.11
kernel module pata_hpt37x low-level driver for the Highpoint HPT37x/30x: 0.6.23
kernel module pata_hpt3x2n low-level driver for the Highpoint HPT3xxN: 0.3.15
kernel module pata_hpt3x3 low-level driver for the Highpoint HPT343/363: 0.6.1
kernel module pata_it821x low-level driver for the IT8211/IT8212 IDE RAID controller: 0.4.2
kernel module pata_jmicron SCSI low-level driver for Jmicron PATA ports: 0.1.5
kernel module pata_marvell SCSI low-level driver for Marvell ATA in legacy mode: 0.1.6
kernel module pata_mpiix low-level driver for Intel MPIIX: 0.7.7
kernel module pata_netcell SCSI low-level driver for Netcell PATA RAID: 0.1.7
kernel module pata_ninja32 low-level driver for Ninja32 ATA: 0.1.5
kernel module pata_ns87410 low-level driver for Nat Semi 87410: 0.4.6
kernel module pata_ns87415 ATA low-level driver for NS87415 controllers: 0.0.1
kernel module pata_oldpiix SCSI low-level driver for early PIIX series controllers: 0.5.5
kernel module pata_pdc2027x libata driver module for Promise PDC20268 to PDC20277: 1.0
kernel module pata_pdc202xx_old low-level driver for Promise 2024x and 20262-20267: 0.4.3
kernel module pata_platform low-level driver for platform device ATA: 1.2
kernel module pata_rdc SCSI low-level driver for RDC PATA controllers: 0.01
kernel module pata_rz1000 low-level driver for RZ1000 PCI ATA: 0.2.4
kernel module pata_sch SCSI low-level driver for Intel SCH PATA controllers: 0.2
kernel module pata_serverworks low-level driver for Serverworks OSB4/CSB5/CSB6: 0.4.3
kernel module pata_sil680 low-level driver for SI680 PATA: 0.4.9
kernel module pata_sis SCSI low-level driver for SiS ATA: 0.5.2
kernel module pata_sl82c105 low-level driver for SI82c105: 0.3.3

kernel module pata_triflex low-level driver for Compaq Triflex: 0.2.8
 kernel module pata_via low-level driver for VIA PATA: 0.3.4
 kernel module pdc_adma Pacific Digital Corporation ADMA low-level driver: 1.0
 kernel module pm80xx PMC-Sierra PM8001/8006/8081/8088/8089/8074/8076/8077/8070/8072 SAS/SATA controller driver: 0.1.38
 kernel module pmcraid PMC Sierra MaxRAID Controller Driver: 1.0.3
 kernel module qed QLogic FastLinQ 4xxxx Core Module: 8.33.0.20
 kernel module qede QLogic FastLinQ 4xxxx Ethernet Driver: 8.33.0.20
 kernel module qedf QLogic QEDF 25/40/50/100Gb FCoE Driver: 8.33.16.20
 kernel module qedi QLogic FastLinQ 4xxxx iSCSI Module: 8.33.0.20
 kernel module qla1280 Qlogic ISP SCSI (qla1x80/qla1x160) driver: 3.27.1
 kernel module qla2xxx QLogic Fibre Channel HBA Driver: 10.00.00.08-k
 kernel module qla3xxx QLogic ISP3XXX Network Driver v2.03.00-k5 : v2.03.00-k5
 kernel module qla4xxx QLogic iSCSI HBA Driver: 5.04.00-k6
 kernel module qlcnlc QLogic 1/10 GbE Converged/Intelligent Ethernet Driver: 5.3.66
 kernel module qlge QLogic 10 Gigabit PCI-E Ethernet Driver : 1.00.00.35
 kernel module r6040 RDC R6040 NAPI PCI FastEthernet driver: 0.29 04Jul2016
 kernel module rsxx IBM Flash Adapter 900GB Full Height Device Driver: 4.0.3.2516
 kernel module s2io : 2.0.26.28
 kernel module sata_dwc_460ex DesignWare Cores SATA controller low level driver: 1.3
 kernel module sata_mv SCSI low-level driver for Marvell SATA controllers: 1.28
 kernel module sata_nv low-level driver for NVIDIA nForce SATA controller: 3.5
 kernel module sata_promise Promise ATA TX2/TX4/TX4000 low-level driver: 2.12
 kernel module sata_qstor Pacific Digital Corporation QStor SATA low-level driver: 0.09
 kernel module sata_sil low-level driver for Silicon Image SATA controller: 2.4
 kernel module sata_sis low-level driver for Silicon Integrated Systems SATA controller: 1.0
 kernel module sata_svw low-level driver for K2 SATA controller: 2.3
 kernel module sata_sx4 Promise SATA low-level driver: 0.12
 kernel module sata_uli low-level driver for ULi Electronics SATA controller: 1.3
 kernel module sata_via SCSI low-level driver for VIA SATA controllers: 2.6
 kernel module sata_vsc low-level driver for Vitesse VSC7174 SATA controller: 2.3
 kernel module sfc Solarflare network driver: 4.1
 kernel module sfc_falcon Solarflare Falcon network driver: 4.1
 kernel module sg SCSI generic (sg) driver: 3.5.36
 kernel module sis190 SiS sis190/191 Gigabit Ethernet driver: 1.4
 kernel module skge SysKonnect Gigabit Ethernet driver: 1.14
 kernel module sky2 Marvell Yukon 2 Gigabit Ethernet driver: 1.30
 kernel module slicoss Alacritech non-accelerated SLIC driver: 1.0
 kernel module smartpqi Driver for Microsemi Smart Family Controller version 1.1.4-130: 1.1.4-130
 kernel module smsc911x : 2008-10-21
 kernel module smsc9420 : 1.01
 kernel module snic Cisco SCSI NIC Driver: 0.0.1.18
 kernel module starfire Adaptec Starfire Ethernet driver: 2.1
 kernel module stex Promise Technology SuperTrak EX Controllers: 6.02.0000.01
 kernel module sunhme Sun HappyMealEthernet(HME) 10/100baseT ethernet driver: 3.10
 kernel module sym53c8xx NCR, Symbios and LSI 8xx and 1010 PCI SCSI adapters: 2.2.3
 kernel module tg3 Broadcom Tigon3 ethernet driver: 3.137
 kernel module thunder_bgx Cavium Thunder BGX/MAC Driver: 1.0
 kernel module thunder_xcv Cavium Thunder RGX/XCV Driver: 1.0
 kernel module tpm TPM Driver: 2.0
 kernel module tpm_atmel TPM Driver: 2.0
 kernel module tpm_crb TPM2 Driver: 0.1
 kernel module tpm_i2c_infineon TPM TIS I2C Infineon Driver: 2.2.0
 kernel module tpm_infineon Driver for Infineon TPM SLD 9630 TT 1.1 / SLB 9635 TT 1.2: 1.9.2
 kernel module tpm_nsc TPM Driver: 2.0

kernel module tpm_st33zp24 ST33ZP24 TPM 1.2 driver: 1.3.0
kernel module tpm_st33zp24_i2c STM TPM 1.2 I2C ST33 Driver: 1.3.0
kernel module tpm_tis TPM Driver: 2.0
kernel module tpm_tis_core TPM Driver: 2.0
kernel module tpm_vtpm_proxy vTPM Driver: 0.1
kernel module tulip Digital 21*4* Tulip ethernet driver: 1.1.15
kernel module typhoon 3Com Typhoon Family (3C990, 3CR990, and variants): 1.0
kernel module ufshcd_core Generic UFS host controller driver Core: 0.2
kernel module virtio_pci virtio-pci: 1
kernel module vmw_pvscsi VMware PVSCSI driver: 1.0.7.0-k
kernel module vmxnet3 VMware vmxnet3 virtual NIC driver: 1.4.16.0-k
kernel module vxlan Driver for VXLAN encapsulated traffic: 0.1
kernel module winbond_840 Winbond W89c840 Ethernet driver: 1.01-e
adduser version: 3.118
apt version: 1.8.2
apt-utils version: 1.8.2
base-files version: 10.3+deb10u4
base-passwd version: 3.5.46
bash version: 5.0-4
bsdmainutils version: 11.1.2+b1
bsdutils version: 1:2.33.1-0.1
busybox version: 1:1.30.1-4
bzip2 version: 1.0.6-9.2~deb10u1
ca-certificates version: 20190110
coreutils version: 8.30-3
cpio version: 2.12+dfsg-9
cron version: 3.0pl1-134+deb10u1
cryptsetup-bin version: 2:2.1.0-5+deb10u2
curl version: 7.64.0-4+deb10u1
dash version: 0.5.10.2-5
dbus version: 1.12.16-1
debconf version: 1.5.71
debconf-i18n version: 1.5.71
debian-archive-keyring version: 2019.1
debiantools version: 4.8.6.1
diffutils version: 1:3.7-3
dirmngr version: 2.2.12-1+deb10u1
dmidecode version: 3.2-1
dmsetup version: 2:1.02.155-3
dosfstools version: 4.1-2
dpkg version: 1.19.7
e2fsprogs version: 1.44.5-1+deb10u3
ethtool version: 1:4.19-1
fdisk version: 2.33.1-0.1
file version: 1:5.35-4+deb10u1
findutils version: 4.6.0+git+20190209-2
gcc-8-base version: 8.3.0-6
gdbm-l10n version: 1.18.1-4
gdisk version: 1.0.3-1.1
gnupg version: 2.2.12-1+deb10u1
gnupg-l10n version: 2.2.12-1+deb10u1
gnupg-utils version: 2.2.12-1+deb10u1
gpg version: 2.2.12-1+deb10u1
gpg-agent version: 2.2.12-1+deb10u1

gpg-wks-client version: 2.2.12-1+deb10u1
gpg-wks-server version: 2.2.12-1+deb10u1
gpgconf version: 2.2.12-1+deb10u1
gpgsm version: 2.2.12-1+deb10u1
gpgv version: 2.2.12-1+deb10u1
grep version: 3.3-1
groff-base version: 1.22.4-3
guile-2.2-libs version: 2.2.4+1-2+deb10u1
gzip version: 1.9-3
hdparm version: 9.58+ds-1
hostname version: 3.21
hwdm version: 21.63-3
ifenslave version: 2.9
ifenslave-2.6 version: 2.9
ifupdown version: 0.8.35
init version: 1.56+nmu1
init-system-helpers version: 1.56+nmu1
initramfs-tools version: 0.133+deb10u1
initramfs-tools-core version: 0.133+deb10u1
iproute2 version: 4.20.0-2
iptables version: 1.8.2-4
iputils-ping version: 3:20180629-2+deb10u1
irqbalance version: 1.5.0-3
isc-dhcp-client version: 4.4.1-2
isc-dhcp-common version: 4.4.1-2
klibc-utils version: 2.0.6-1
kmod version: 26-1
kpartx version: 0.7.9-3
krb5-locales version: 1.17-3
less version: 487-0.1+b1
libacl1 version: 2.2.53-4
libaio1 version: 0.3.112-3
libapparmor1 version: 2.13.2-10
libapt-inst2.0 version: 1.8.2
libapt-pkg5.0 version: 1.8.2
libargon2-1 version: 0~20171227-0.2
libassuan0 version: 2.5.2-1
libattr1 version: 1:2.4.48-4
libaudit-common version: 1:2.8.4-3
libaudit1 version: 1:2.8.4-3
libblkid1 version: 2.33.1-0.1
libboost-atomic1.67.0 version: 1.67.0-13+deb10u1
libboost-python1.67.0 version: 1.67.0-13+deb10u1
libboost-system1.67.0 version: 1.67.0-13+deb10u1
libboost-thread1.67.0 version: 1.67.0-13+deb10u1
libbsd0 version: 0.9.1-2
libbz2-1.0 version: 1.0.6-9.2~deb10u1
libc-bin version: 2.28-10
libc6 version: 2.28-10
libcap-ng0 version: 0.7.9-2
libcap2 version: 1:2.25-2
libcap2-bin version: 1:2.25-2
libcom-err2 version: 1.44.5-1+deb10u3
libcryptsetup12 version: 2:2.1.0-5+deb10u2

libcurl4 version: 7.64.0-4+deb10u1
libdb5.3 version: 5.3.28+dfsg1-0.5
libdbus-1-3 version: 1.12.16-1
libdebconfclient0 version: 0.249
libdevmapper1.02.1 version: 2:1.02.155-3
libdns-export1104 version: 1:9.11.5.P4+dfsg-5.1
libedit2 version: 3.1-20181209-1
libelf1 version: 0.176-1.1
libestr0 version: 0.1.10-2.1
libevent-core-2.1-6 version: 2.1.8-stable-4
libevent-pthreads-2.1-6 version: 2.1.8-stable-4
libexpat1 version: 2.2.6-2+deb10u1
libext2fs2 version: 1.44.5-1+deb10u3
libfastjson4 version: 0.99.8-2
libfdisk1 version: 2.33.1-0.1
libffi6 version: 3.2.1-9
libfribidi0 version: 1.0.5-3.1+deb10u1
libgc1c2 version: 1:7.6.4-0.4
libgcc1 version: 1:8.3.0-6
libgcrypt20 version: 1.8.4-5
libgdbm-compat4 version: 1.18.1-4
libgdbm6 version: 1.18.1-4
libglib2.0-0 version: 2.58.3-2+deb10u2
libglib2.0-data version: 2.58.3-2+deb10u2
libgmp10 version: 2:6.1.2+dfsg-4
libgnutls-openssl27 version: 3.6.7-4+deb10u3
libgnutls30 version: 3.6.7-4+deb10u3
libgpg-error0 version: 1.35-1
libgsasl7 version: 1.8.0-8+b2
libgssapi-krb5-2 version: 1.17-3
libhd21 version: 21.63-3
libhogweed4 version: 3.4.1-1
libicu63 version: 63.1-6+deb10u1
libidn11 version: 1.33-2.2
libidn2-0 version: 2.0.5-1+deb10u1
libip4tc0 version: 1.8.2-4
libip6tc0 version: 1.8.2-4
libiptc0 version: 1.8.2-4
libisc-export1100 version: 1:9.11.5.P4+dfsg-5.1
libjson-c3 version: 0.12.1+ds-2
libk5crypto3 version: 1.17-3
libkeyutils1 version: 1.6-6
libklibc version: 2.0.6-1
libkmod2 version: 26-1
libkrb5-3 version: 1.17-3
libkrb5support0 version: 1.17-3
libksba8 version: 1.3.5-2
libkyotocabinet16v5 version: 1.2.76-4.2+b1
libldap-2.4-2 version: 2.4.47+dfsg-3+deb10u2
libldap-common version: 2.4.47+dfsg-3+deb10u2
liblocale-gettext-perl version: 1.07-3+b4
liblognorm5 version: 2.0.5-1
libltdl7 version: 2.4.6-9
liblz4-1 version: 1.8.3-1

liblzma5 version: 5.2.4-1
liblzo2-2 version: 2.10-0.1
libmagic-mgc version: 1:5.35-4+deb10u1
libmagic1 version: 1:5.35-4+deb10u1
libmailutils5 version: 1:3.5-3
libmariadb3 version: 1:10.3.22-0+deb10u1
libmnl0 version: 1.0.4-2
libmount1 version: 2.33.1-0.1
libmpdec2 version: 2.4.2-2
libncurses6 version: 6.1+20181013-2+deb10u2
libncursesw6 version: 6.1+20181013-2+deb10u2
libnetfilter-contrack3 version: 1.0.7-1
libnettle6 version: 3.4.1-1
libnewt0.52 version: 0.52.20-8
libnfnetwork0 version: 1.0.1-3+b1
libnftnl11 version: 1.1.2-2
libnghttp2-14 version: 1.36.0-2+deb10u1
libnpt0 version: 1.6-1
libntlm0 version: 1.5-1
libnuma1 version: 2.0.12-1
libopenipmi0 version: 2.0.25-2.1
libopts25 version: 1:5.18.12-4
libp11-kit0 version: 0.23.15-2
libpam-modules version: 1.3.1-5
libpam-modules-bin version: 1.3.1-5
libpam-runtime version: 1.3.1-5
libpam-systemd version: 241-7~deb10u4
libpam0g version: 1.3.1-5
libpcap0.8 version: 1.8.1-6
libpci3 version: 1:3.5.2-1
libpcre3 version: 2:8.39-12
libperl5.28 version: 5.28.1-6
libpopt0 version: 1.16-12
libprocps7 version: 2:3.3.15-2
libpsl5 version: 0.20.2-2
libpython2.7 version: 2.7.16-2+deb10u1
libpython2.7-minimal version: 2.7.16-2+deb10u1
libpython2.7-stdlib version: 2.7.16-2+deb10u1
libpython3.7 version: 3.7.3-2+deb10u1
libpython3.7-minimal version: 3.7.3-2+deb10u1
libpython3.7-stdlib version: 3.7.3-2+deb10u1
libreadline7 version: 7.0-5
librtmp1 version: 2.4+20151223.gitfa8646d.1-2
libsasl2-2 version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules-db version: 2.1.27+dfsg-1+deb10u1
libseccomp2 version: 2.3.3-4
libselinux1 version: 2.8-1+b1
libsemanage-common version: 2.8-2
libsemanage1 version: 2.8-2
libsensors-config version: 1:3.5.0-3
libsensors5 version: 1:3.5.0-3
libsepol1 version: 2.8-1
libsgutils2-2 version: 1.44-1

libslang2 version: 2.3.2-2
libsmartcols1 version: 2.33.1-0.1
libsnmp-base version: 5.7.3+dfsg-5
libsnmp30 version: 5.7.3+dfsg-5
libsqlite3-0 version: 3.27.2-3
libss2 version: 1.44.5-1+deb10u3
libssh2-1 version: 1.8.0-2.1
libssl1.1 version: 1.1.1d-0+deb10u3
libstdc++6 version: 8.3.0-6
libsysfs2 version: 2.1.0+repack-5
libsystemd0 version: 241-7~deb10u4
libtasn1-6 version: 4.13-3
libtext-charwidth-perl version: 0.04-7.1+b1
libtext-iconv-perl version: 1.7-5+b7
libtext-wrapi18n-perl version: 0.06-7.1
libtinfo6 version: 6.1+20181013-2+deb10u2
libuchardet0 version: 0.0.6-3
libudev1 version: 241-7~deb10u4
libunistring2 version: 0.9.10-1
liburcu6 version: 0.10.2-1
libuuid1 version: 2.33.1-0.1
libwrap0 version: 7.6.q-28
libx11-6 version: 2:1.6.7-1
libx11-data version: 2:1.6.7-1
libx86emu2 version: 2.0-1
libxau6 version: 1:1.0.8-1+b2
libxcb1 version: 1.13.1-2
libxdmcp6 version: 1:1.1.2-3
libxext6 version: 2:1.3.3-1+b2
libxml2 version: 2.9.4+dfsg1-7+b3
libxmuu1 version: 2:1.1.2-2+b3
libxtables12 version: 1.8.2-4
libyajl2 version: 2.1.0-3
libzstd1 version: 1.3.8+dfsg-3
linux-base version: 4.6
linux-firmware version: 1.183.2
login version: 1:4.5-1.1
logrotate version: 3.14.0-4
lsb-base version: 10.2019051400
lsscsi version: 0.30-0.1
mailutils version: 1:3.5-3
mailutils-common version: 1:3.5-3
mariadb-common version: 1:10.3.22-0+deb10u1
mawk version: 1.3.3-17+b3
mime-support version: 3.62
mount version: 2.33.1-0.1
multipath-tools version: 0.7.9-3
multipath-tools-boot version: 0.7.9-3
mysql-common version: 5.8+1.0.5
nano version: 3.2-3
ncurses-base version: 6.1+20181013-2+deb10u2
ncurses-bin version: 6.1+20181013-2+deb10u2
ncurses-term version: 6.1+20181013-2+deb10u2
net-tools version: 1.60+git20180626.aebd88e-1

netbase version: 5.6
nload version: 0.7.4-2+b1
ntp version: 1:4.2.8p12+dfsg-4
numactl version: 2.0.12-1
openipmi version: 2.0.25-2.1
openssh-client version: 1:7.9p1-10+deb10u2
openssh-server version: 1:7.9p1-10+deb10u2
openssh-sftp-server version: 1:7.9p1-10+deb10u2
openssl version: 1.1.1d-0+deb10u3
passwd version: 1:4.5-1.1
perl version: 5.28.1-6
perl-base version: 5.28.1-6
perl-modules-5.28 version: 5.28.1-6
pigz version: 2.4-1
pinentry-curses version: 1.1.0-2
powermgmt-base version: 1.34
procps version: 2:3.3.15-2
publicsuffix version: 20190415.1030-1
python3.7 version: 3.7.3-2+deb10u1
python3.7-minimal version: 3.7.3-2+deb10u1
qemu-guest-agent version: 1:3.1+dfsg-8+deb10u5
readline-common version: 7.0-5
rsyslog version: 8.1901.0-1
runit-helper version: 2.8.6
sdparm version: 1.10-1
sed version: 4.7-1
sensible-utils version: 0.0.12
sg3-utils version: 1.44-1
sg3-utils-udev version: 1.44-1
shared-mime-info version: 1.10-1
smartmontools version: 6.6-1
smp-utils version: 0.98-2
snmp version: 5.7.3+dfsg-5
snmpd version: 5.7.3+dfsg-5
snmp version: 1:4.2.8p12+dfsg-4
ssmtp version: 2.64-8.1
sysstat version: 12.0.3-2
systemd version: 241-7~deb10u4
systemd-sysv version: 241-7~deb10u4
sysvinit-utils version: 2.93-8
tar version: 1.30+dfsg-6
tasksel version: 3.53
tasksel-data version: 3.53
tcpdump version: 4.9.3-1~deb10u1
tofrodos version: 1.7.13+ds-4
traceroute version: 1:2.1.0-2
tzdata version: 2020a-0+deb10u1
ucf version: 3.0038+nmu1
udev version: 241-7~deb10u4
util-linux version: 2.33.1-0.1
vim-common version: 2:8.1.0875-5
vim-tiny version: 2:8.1.0875-5
whiptail version: 0.52.20-8
xauth version: 1:1.0.10-1

xdg-user-dirs version: 0.17-2
xxd version: 2:8.1.0875-5
xz-utils version: 5.2.4-1
zlib1g version: 1:1.2.11.dfsg-1
linux-firmware: 1.183.2
ssmtp: 2.64-8.1

Elasticsearch-specific and additional Caringo distributions:

Elasticsearch 6.8.6 / 5.6.12 / 2.3.3

elasticsearch-curator 4.3.1 (supports Elasticsearch 5 and 6)

txes 0.1.4+

Swarm S3 Backup Restore 1.1.1

Swarm Search 6.3.1

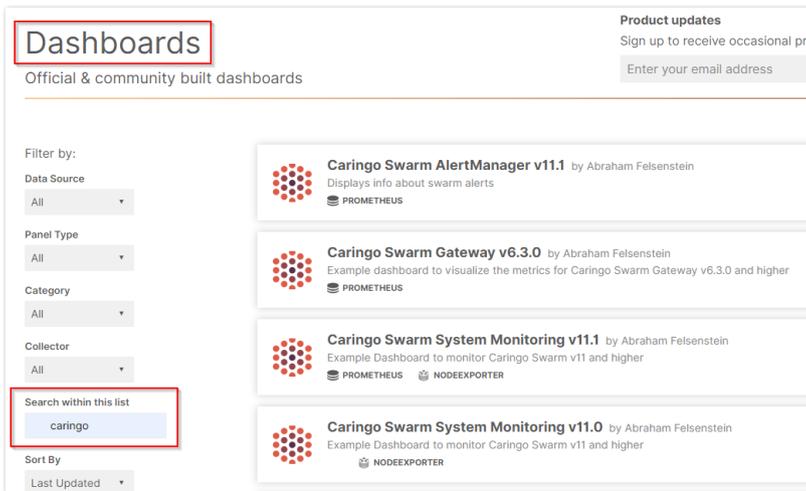
Swarm Metrics 6.3.1

Swarm Storage 11.1 Release

- [New Features](#)
- [Additional Changes](#)
- [Upgrade Impacts](#)
- [Watch Items and Known Issues](#)
- [Upgrading Swarm](#)

New Features

Grafana Dashboards for Swarm Monitoring – To offer sophisticated visualization of the Prometheus Node Exporter and related Swarm data, DataCore publishes public Grafana dashboards for monitoring Swarm implementations. To see the latest dashboards for Swarm products and features, search the dashboards for Caringo: <https://grafana.com/grafana/dashboards?search=caringo>. See [Prometheus Node Exporter and Grafana](#).



Customized dashboards are available for **Swarm System Monitoring**, with separate dashboards for specific to versions of Swarm Storage, starting with 10.2. The detailed dashboard covers cluster health, capacity, indexing, licensing, temperature, and network and CPU loads, as well as cluster-wide operations:



- The Prometheus Node Exporter produces a totaled version of each of the SCSP-related statistics (appending `_total` to the original name), to capture counts in addition to aggregate rates. These totaled statistics for Swarm HTTP operations and responses are incorporated into the Grafana dashboard for Swarm 11.1. (SWAR-8710)

Elasticsearch 6 – Swarm supports and ships with Elasticsearch 6, which is a version allowing upgrades-in-place (without reindexing) going forward several releases. Both ES2 and ES5 is deprecated in the next release. Create a new ES6 cluster, add a new search feed (to reindex metadata), and switch over to it when the reindexing is complete to migrate from either ES2 or ES5. See [Migrating from Older Elasticsearch](#).

Upgrade to Python 3 – All Swarm Storage usage of Python 2 is uniformly upgraded to Python 3, which brings a small performance boost, up to 20% improvement for high loads. (SWAR-8143)

Modernization – Extensive work has modernized the Linux kernel to Debian 10 and its drivers and components, which allowed for comprehensive updates across Swarm's third-party tools and dependencies. See [Third-Party Components for 11.1](#) for the complete listing. (SWAR-8664)

Administration Improvements – This release includes several changes to make it easier to monitor and manage Swarm:

- Swarm has improved handling of slashes in object naming to prevent unintended naming and renaming errors. Leading slashes are always removed, and trailing spaces are removed from bucket names. Trailing slashes in domain names cause 404 errors, but trailing slashes are valid for named objects, so they are retained. (SWAR-8706)
- Multipart writes are long-running operations that provide an initial 202 Accepted response and a later 201 Created response, on completion. For S3 compatibility, the initial response now includes a Completion-ETag with the value of the expected ETag. If there is an error, there is no new object, and the expected ETag provided is not valid. (SWAR-8694)
- For a multipart object, to copy from a start range to the end of the object, do so by omitting the range end. This avoids the risk of the end value extending beyond the size of the object being copied, which results in a 416 Range Not Satisfiable response. (SWAR-8675)
- Logging of disk diagnostics (such as dmesg and SMART data) now covers volume retires that are due to I/O device errors, in addition to volume failures. (SWAR-8665)
- Swarm 11.1 has improved volume health monitoring and alerting to surface overly long I/O request times that may be an indication of a volume nearing its end of life. (SWAR-8585)
- When returning a list of drives via the management API (/api/storage/chassis/*/drives), Swarm now returns both the drive name (such as /dev/sdd) and the volume's UUID. (SWAR-8637)
- Replication feed handling now generates more accurate state reporting and helpful status descriptions, to support diagnosis of blocked feeds. (SWAR-8660)
- All Swarm Management API endpoints that required specifying the cluster name now accept "_self" to refer to the local cluster, which eases formation of the call. (SWAR-8636)
- Error messaging now clarifies when an attempt to update a Swarm setting via the API has failed because the setting is read-only. (SWAR-8443)
- Swarm no longer ignores erroneous use of the "format" query argument on a non-listing request (a request other than GET or HEAD). Swarm now returns a 400 Bad Request error. (SWAR-8598)
- The retired setting `cluster.settingsUuid` is now ignored by Swarm, which guarantees obsolete values do not prevent Swarm from booting. (SWAR-8535)

Additional Changes

These items are other changes and improvements including those that come from testing and user feedback.

- **OSS Versions** – See [Third-Party Components for 11.1](#) for the complete listing of packages and versions.
 - The Linux kernel is upgraded to 4.19.84. (SWAR-8664)
 - Linux firmware is upgraded to 1.183.2. (SWAR-8664)
- **Fixed in 11.1.0**
 - Persisted settings, including security.administrators, may not update properly when the persisted settings object was read at startup. This issue mostly affected chassis with encrypted volumes or more than 6 volumes. (SWAR-8800)
 - With Elasticsearch 5, listing a bucket or domain with fields=all and format=json receives a response with invalid JSON. (SWAR-8781)
 - Premature closes of EC object reads sometimes cause abnormal memory usage and critical errors. (SWAR-8709)
 - Read failures (500: ZeroDivisionError) can occur with small range reads near the end of EC objects, for certain encodings. (SWAR-8661)
 - In versions 10.x-11.0 are used with ES 5.6, deprecation warnings caused logs to consume excessive disk space. (SWAR-8632)
 - Unnamed objects can appear in listings even after they are deleted. (SWAR-8623)
 - Under some conditions, Swarm may start without mounting some of its volumes. (SWAR-8597)

Upgrade Impacts

Use the supported versions of Swarm components for the target version of Elasticsearch:

Elasticsearch 6.8.6	Swarm Storage 11.1	Gateway 6.3	SwarmFS 2.4	Recommended configuration.
Elasticsearch 5.6.12	Swarm Storage 10.0 - 11.1	Gateway 6.0 - 6.3	SwarmFS 2.4	Plan to migrate to Elasticsearch 6 . Support for earlier versions is ending.
Elasticsearch 2.3.3	Swarm Storage 9.6 - 11.1	Gateway 5.4	SwarmFS 2.1	

These items are changes to the product function that may require operational or development changes for integrated applications. Address the upgrade impacts for each of the versions since the one currently running.

Impacts for 11.1

- Upgrading Elasticsearch** – Use Elasticsearch 5.6.12/2.3.3 with Storage 11.1 if moving to ES 6 immediately is not possible, but start the migration now (see [Migrating from Older Elasticsearch](#)). Support for ES 5.6.12/2.3.3 ends in a future release, and testing for 2.3.3 with Swarm 11 is discontinued. *Important:* Always upgrade Swarm Search and Metrics at the same time ES is upgraded. Do not run an ES 5 Search or Metrics Curator against ES 6.
- Swarm Search and Metrics** – This release includes new versions of Swarm Search and Metrics RPMs. Both require Python 3 to be installed on the ES servers they run on.
 - For Swarm Metrics on RHEL/CentOS 7.7, first install this dependency: `yum install epel-release`
- Python 3** – Install Python 3 if is not automatically installed with RHEL/CentOS 7.
- Propagate Delete Removed** – For [replication and S3 backup feeds](#), the Propagate Deletes option is removed from the legacy Admin Console and the Management API (`propagateDeletes`, `nodeDeletes` fields). (SWAR-8609, SWAR-8615)
- Swarm Configuration** – Run the [Storage Settings Checker](#) before upgrading to this version, to identify configuration issues.
 - The Storage Settings Checker now requires Python 3 to be installed. (SWAR-8742)
 - `crier.deadVolumeWall` has been unpublished for reimplementaion. (SWAR-8640)
- S3 Backup Restore** – The S3 Backup Restore Tool has been migrated to Python 3.6. If the tool is installed, uninstall it and [install the new version](#). (SWAR-8703)
- Upgrade Process** – During the upgrade to 11.1, it may not be possible to monitor the cluster via the Swarm UI. Workaround: Use the legacy Admin Console (port 90) during upgrade. (SWAR-8716)
- Differences in `scsp.forceLegacyNonce` configuration** depending on the version being upgraded from (SWAR-9020):
- If currently running a Swarm Storage version prior to 11.1**, and upgrading to 11.1, 11.2, 11.3, 12.0 or 12.1:

Before upgrading, set `scsp.forceLegacyNonce=true` in the `node.cfg` file. After the upgrade, when the cluster is fully up, update `scsp.forceLegacyNonce=false` using `swarmctl` and change `scsp.forceLegacyNonce=false` in the `node.cfg` file.

If currently running a Swarm Storage version 11.1, 11.2, 11.3, 12.0 or 12.1 and upgrading to another version from that list:

Before upgrading, verify `scsp.forceLegacyNonce=false` is in the `node.cfg` file and verify using `swarmctl` that `scsp.forceLegacyNonce=false` in the cluster.

Use `swarmctl` to check or change settings

Use `'swarmctl -C scsp.forceLegacyNonce'` to check the value of `scsp.forceLegacyNonce`.
 Use `'swarmctl -C scsp.forceLegacyNonce -V False'` to set the value to `false`.

For more details, see <https://support.cloud.caringo.com/tools/Tech-Support-Scripts-Bundle-swarmctl.pdf>.

Impacts for 11.0

- **Upgrading Elasticsearch** – You may use Elasticsearch 2.3.3 with Storage 11.0 if you cannot move to 5.6 now, but plan the migration immediately (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release, and testing with Swarm 11 is discontinuing.
- **Propagate Deletes Deprecated** – The option to disable *Propagate Deletes* on [replication feeds](#) is deprecated; use [Object Versioning](#) to preserve deleted content. Do not disable Propagate Deletes when versioning is enabled or when defining an S3 Backup. (SWAR-8609)
- **Configuration Settings** – Run the [Storage Settings Checker](#) to identify these and other configuration issues.
 - Changed settings:
 - `ec.segmentConsolidationFrequency` (`ecSegmentConsolidationFrequency` in SNMP) has an improved default (10), which you must apply to your cluster when you upgrade. (SWAR-8483)
 - `cluster.name` is now required. Add it to the `cluster.cfg` file. (SWAR-8466).
 - `metrics.nodeExporterFrequency` (`metricsExporterFrequency` in SNMP) is now a persisted cluster setting. (SWAR-8467).
 - Removed settings:
 - `chassis.processes` is allowed but is ignored.
 - Numerous settings are now promoted to *cluster-level* (versus node-level) scope, so you can manage them via **Settings > Cluster** in the Swarm UI (SWAR-8457):
 - `console.expiryErrInterval`
 - `console.expiryWarnInterval`
 - `console.indexErrorLevel`
 - `console.indexWarningLevel`
 - `console.port`
 - `console.reportStyleUrl`
 - `console.spaceErrorLevel`
 - `console.spaceWarnLevel`
 - `console.styleUrl`
 - `feeds.retry`
 - `feeds.statsReportInterval`
 - `health.parallelWriteTimeout`
 - `log.obscureUUIDs`
 - `metrics.enableNodeExporter`
 - `network.dnsDomain`
 - `network.dnsServers`
 - `network.icmpAcceptRedirects`
 - `network.igmpVersion`
 - `network.mtu`
 - `startup.certificates`

Impacts for 10.2

- **Upgrading Elasticsearch** – You may continue to use Elasticsearch 2.3.3 with Storage 10.2 until you are able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release. the upgrade to Elasticsearch 5.6 must be completed before [upgrading to Gateway 6.0](#).
- **Configuration Settings** – Run the [Storage Settings Checker](#) before any Swarm 10 upgrade to identify configuration issues. Note these changes:
 - `ec.protectionLevel` is now persisted. (SWAR-8231)
 - `index.ovMinNodes=3` is the new default for the overlay index, in support of Swarm 10's new architecture. To keep your overlay index operational, set this new value in your cluster, through the UI or by SNMP (overlayMinNodes). (SWAR-8278)
 - `metrics.enableNodeExporter` can be set to True, which enables the Prometheus Node Exporter on that node. (SWAR-8408, SWAR-8578)
 - `metrics.nodeExporterFrequency`, a new dynamic setting, sets how frequently to refresh Swarm-specific Prometheus metrics in Elasticsearch; it defaults to 0, which disables this export. (SWAR-8408).

Impacts for 10.1

- **Upgrading Elasticsearch** – Continue to use Elasticsearch 2.3.3 with Storage 10.1 until able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release. Complete the upgrade to Elasticsearch 5.6 before upgrading to Gateway 6.0.
- **Configuration Settings** – Run the [Storage Settings Checker](#) before any Swarm 10 upgrade to identify configuration issues.
 - `metrics.enableNodeExporter=true` enables Swarm to run the Prometheus node exporter on port 9100. (SWAR-8170)
- **IP address update delay** – When upgrading from Swarm 9 to the new architecture of Swarm 10, note the "ghosts" of previously used IP addresses may appear in the Storage UI; these resolve within 4 days. (SWAR-8351)
- **Update MIBs on CSN** – Before upgrading to Storage 10.x, the MIBs on the CSN must be updated. From the Swarm Support tools bundle, run the `platform-update-mibs.sh` script. (CSN-1872)

Impacts for 10.0

- **Upgrading Elasticsearch** – You may continue to use Elasticsearch 2.3.3 with Storage 10.0 until you are able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release.
- **Configuration Settings** – Run the [Storage Settings Checker](#) to identify these and other configuration issues.
 - Changes for the new single-IP dense architecture:
 - `network.ipAddress` – multiple IP addresses now disallowed
 - `chassis.processes` – removed; multi-server configurations are no longer supported
 - `ec.protectionLevel` – new value "volume"
 - `ec.subclusterLossTolerance` – removed
 - Changes for security (see next section)
 - `security.administrators`, `security.operators` – removed 'snmp' user
 - `snmp.rwCommunity`, `snmp.roCommunity` – new settings for 'snmp' user
 - `startup.certificates` – new setting to hold any and all public keys
 - New settings:
 - `disk.atimeEnabled`
 - `health.parallelWriteTimeout`
 - `search.pathDelimiter`

- **Required SNMP security change** – Remove the `snmp` key from the `security.administrators` setting, and update `snmp.rwCommunity` with its value. Nodes that contain only the `snmp` key in the `security.administrators` setting does not boot. If you changed the default value of the `snmp` key in the `security.operators` setting, update `snmp.roCommunity` with that value and then remove the `snmp` key from `security.operators`. In the `security.operators` setting, 'snmp' is a reserved key, and it cannot be an authorized console operator name. (SWAR-8097)
- **EC protection**
 - *Best practice:* Use `ec.protectionLevel=node`, which distributes segments across the cluster's physical/virtual machines. Do not use `ec.protectionLevel=subcluster` unless you already have subclusters defined and are sure the specified EC encoding is supported. A new level, `ec.protectionLevel=volume`, allows EC writes to succeed if you have a small cluster with fewer than $(k+p)/p$ nodes. (Swarm always seeks the highest protection possible for EC segments, regardless of the level you set.)
 - Optimize hardware for EC by verifying there are more than $k+p$ subclusters/nodes (as set by `ec.protectionLevel`); for example, with `policy.ecEncoding=5:2`, you need at least 8 subclusters/nodes. When Swarm cannot distribute EC segments adequately for protection, EC writes can fail despite ample free space. (SWAR-7985)
 - Setting `ec.protectionLevel=subcluster` without creating subclusters (defining `node.subcluster` across sets of nodes) causes a critical error and lowers the protection level to 'node'. (SWAR-8175)
- **Small clusters** – Verify the following settings if using 10 or fewer Swarm nodes. Do not use fewer than 3 in production. *Important:* If you need to change any, do so *before* upgrading to Swarm 10.
 - **policy.replicas** – The `min` and `default` values for numbers of replicas to keep in your cluster must not exceed your number of nodes. For example, a 3-node cluster may have only `min=2` or `min=3`.
 - **EC encoding and protection** – For EC encoding, verify you have enough nodes to support the cluster's encoding (`policy.ecEncoding`). For EC writes to succeed with fewer than $(k+p)/p$ nodes, use the new level, `ec.protectionLevel=volume`.
 - *Best practice:* Keep at least one physical machine in your cluster beyond the minimum number needed. This allows for one machine to be down for maintenance without compromising the constraint.
- **"Cluster in a box"** – Swarm supports a "cluster in a box" configuration as long as that box is running a virtual machine host and Swarm instances are running in 3 or more VMs. Each VM boots separately and has its own IP address. Follow the recommendations for small clusters, substituting VMs for nodes. If you have two physical machines, use the "cluster in a box" configuration, but move to direct booting of Swarm with 3 or more.
- **Offline node status** – Because Swarm 10's new architecture reduces the number of IP addresses in your storage cluster, you may see the old IPs and subclusters reporting as **Offline** nodes until they timeout in 4 days (`crier.forgetOfflineInterval`), which is expected.

Watch Items and Known Issues

The following operational limitations and watch items exist in this release.

- Infrequent WARNING messages, "Node/Volume entry not published due to lock contention (...); action is retried," may appear in logs. Unless they are frequent, they may be ignored. (SWAR-8802)
- If a node mounts an encrypted volume that is missing the encryption key in the configuration, the node fails to mount all disks in the node. (SWAR-8762)
- S3 Backup feeds do not back up logical objects greater than 5 GB. (SWAR-8554)
- If downgrading from Swarm 11.0, CRITICAL errors may appear on the feeds. To stop the errors, edit the existing feed definition names via the Swarm UI or legacy Admin Console. (SWAR-8543)
- When restarting a cluster of virtual machines that are UEFI-booted (versus legacy BIOS), the chassis shut down but do not come back up. (SWAR-8054)
- If Elasticsearch cluster is wiped, the Storage UI shows no NFS config. Contact DataCore Support for help repopulating the SwarmFS config information. (SWAR-8007)
- If a bucket is deleted, any incomplete multipart upload into that bucket leaves its parts (unnamed streams) in the domain. To find and delete them, use the `s3cmd` utility (search the Support site for "s3cmd" for guidance). (SWAR-7690)

- Logs showed the error "FEEDS WARNING: calcFeedInfo(etag=xxx) cannot find domain xxx, which is needed for a domains-specific replication feed". The root cause is fixed; if receiving such warnings, contact DataCore Support so the issue can be resolved. (SWAR-7556)
- With multipath-enabled hardware, the Swarm console Disk Volume Menu may erroneously show too many disks, having multiplied the actual disks in use by the number of possible paths to them. (SWAR-7248)

Note these installation issues:

- The elasticsearch-curator package may show an error during an upgrade, which is a known curator issue. Workaround: Reinstall the curator: `yum reinstall elasticsearch-curator` (SWAR-7439)
- Do not install the Swarm Search RPM before installing Java. If Gateway startup fails with "Caringo script plugin is missing from indexer nodes", uninstall and reinstall the Swarm Search RPM. (SWAR-7688)

Upgrading Swarm

Proceed to [How to Upgrade Swarm](#) to upgrade Swarm 9 or higher.



Important

Contact DataCore Support for guidance if needing to upgrade from Swarm 8.x or earlier.

Third-Party Components for 11.1

zope.interface version: 4.7.1
ipaddress version: 1.0.23
cryptography version: 2.8
pyOpenSSL version: 19.1.0
service_identity version: 18.1.0
incremental version: 17.5.0
Twisted[tls] version: 19.10.0
pyutil version: 3.3.0
python-dateutil version: 2.8.1
Werkzeug version: 0.16.0
klein version: 19.6.0
requests version: 2.21.0
zfec version: 1.5.3
yajl-py version: 2.1.2
certifi version: *latest as of 2020-04-06*
pyratemp version: 0.3.2
Newt version: 0.52.20
Prometheus node_exporter version: 0.18.1
libpng version: 1.2.8
LILO version: 22.7.1
Operating system: Debian GNU/Linux 10 (buster)
Linux kernel: 4.19.84
kernel module 3w_9xxx 3ware 9000 Storage Controller Linux Driver: 2.26.02.014
kernel module 3w_sas LSI 3ware SAS/SATA-RAID Linux Driver: 3.26.02.000
kernel module 3w_xxxx 3ware Storage Controller Linux Driver: 1.26.02.003
kernel module 8021q : 1.8
kernel module 8139cp RealTek RTL-8139C+ series 10/100 PCI Ethernet driver: 1.3
kernel module 8139too RealTek RTL-8139 Fast Ethernet driver: 0.9.28
kernel module aacraid Dell PERC2, 2/Si, 3/Si, 3/Di, Adaptec Advanced Raid Products, HP NetRAID-4M, IBM ServeRAID & ICP SCSI driver: 1.2.1
[50877]-custom
kernel module acard_ahci ACard AHCI SATA low-level driver: 1.0
kernel module ad7418 AD7416/17/18 driver: 0.4
kernel module ahci AHCI SATA low-level driver: 3.0
kernel module aic79xx Adaptec AIC790X U320 SCSI Host Bus Adapter driver: 3.0
kernel module aic7xxx Adaptec AIC77XX/78XX SCSI Host Bus Adapter driver: 7.0
kernel module aic94xx Adaptec aic94xx SAS/SATA driver: 1.0.3
kernel module am53c974 AM53C974 SCSI driver: 1.00
kernel module amd_xgbe AMD 10 Gigabit Ethernet Driver: 1.0.3
kernel module arcmsr Areca ARC11xx/12xx/16xx/188x SAS/SATA RAID Controller Driver: v1.40.00.09-20180709
kernel module ata_generic low-level driver for generic ATA: 0.2.15
kernel module ata_piix SCSI low-level driver for Intel PIIX/ICH ATA controllers: 2.13
kernel module atl1 Atheros L1 Gigabit Ethernet Driver: 2.1.3
kernel module atl1c Qualcomm Atheros 100/1000M Ethernet Network Driver: 1.0.1.1-NAPI
kernel module atl1e Atheros 1000M Ethernet Network Driver: 1.0.0.7-NAPI
kernel module atl2 Atheros Fast Ethernet Network Driver: 2.2.3
kernel module atlantic aQuantia Corporation(R) Network Driver: 2.0.3.0-kern
kernel module atxp1 System voltages control via Attansic ATXP1: 0.6.3
kernel module b44 Broadcom 44xx/47xx 10/100 PCI ethernet driver: 2.0
kernel module be2iscsi Emulex OneConnectOpen-iSCSI Driver version11.4.0.1 Driver 11.4.0.1: 11.4.0.1
kernel module be2net Emulex OneConnect NIC Driver 12.0.0.0: 12.0.0.0

kernel module bfa QLogic BR-series Fibre Channel HBA Driver fcpim: 3.2.25.1
kernel module bna QLogic BR-series 10G PCIe Ethernet driver: 3.2.25.1
kernel module bnx2 QLogic BCM5706/5708/5709/5716 Driver: 2.2.6
kernel module bnx2fc QLogic FCoE Driver: 2.11.8
kernel module bnx2i QLogic NetXtreme II BCM5706/5708/5709/57710/57711/57712/57800/57810/57840 iSCSI Driver: 2.7.10.1
kernel module bnx2x QLogic BCM57710/57711/57711E/57712/57712_MF/57800/57800_MF/57810/57810_MF/57840/57840_MF Driver: 1.712.30-0
kernel module bnxt_en Broadcom BCM573xx network driver: 1.9.2
kernel module bonding Ethernet Channel Bonding Driver, v3.7.1: 3.7.1
kernel module cnic QLogic cnic Driver: 2.5.22
kernel module csiostor Chelsio FCoE driver: 1.0.0-ko
kernel module cxgb3 Chelsio T3 Network Driver: 1.1.5-ko
kernel module cxgb3i Chelsio T3 iSCSI Driver: 2.0.1-ko
kernel module cxgb4 Chelsio T4/T5/T6 Network Driver: 2.0.0-ko
kernel module cxgb4i Chelsio T4-T6 iSCSI Driver: 0.9.5-ko
kernel module cxgb4vf Chelsio T4/T5/T6 Virtual Function (VF) Network Driver: 2.0.0-ko
kernel module dca : 1.12.1
kernel module dcdbas Dell Systems Management Base Driver (version 5.6.0-3.2): 5.6.0-3.2
kernel module de2104x Intel/Digital 21040/1 series PCI Ethernet driver: 0.7
kernel module dmfe Davicom DM910X fast ethernet driver: 1.36.4
kernel module e100 Intel(R) PRO/100 Network Driver: 3.5.24-k2-NAPI
kernel module e1000 Intel(R) PRO/1000 Network Driver: 7.3.21-k8-NAPI
kernel module e1000e Intel(R) PRO/1000 Network Driver: 3.2.6-k
kernel module eeprom_93cx6 EEPROM 93cx6 chip driver: 1.0
kernel module efivars sysfs interface to EFI Variables: 0.08
kernel module ena Elastic Network Adapter (ENA): 1.5.0K
kernel module enic Cisco VIC Ethernet NIC Driver: 2.3.0.53
kernel module esas2r esas2r: 1.00
kernel module esp_scsi ESP SCSI driver core: 2.000
kernel module fm10k Intel(R) Ethernet Switch Host Interface Driver: 0.23.4-k
kernel module fnic Cisco FCoE HBA Driver: 1.6.0.34
kernel module hpsa Driver for HP Smart Array Controller version 3.4.20-125: 3.4.20-125
kernel module i40e Intel(R) 40-10 Gigabit Ethernet Connection Network Driver: 2.7.29
kernel module i40e Intel(R) Ethernet Connection XL710 Network Driver: 2.3.2-k
kernel module i40evf Intel(R) XL710 X710 Virtual Function Network Driver: 3.2.2-k
kernel module ice Intel(R) Ethernet Connection E800 Series Linux Driver: ice-0.7.0-k
kernel module igb Intel(R) Gigabit Ethernet Network Driver: 5.4.0-k
kernel module igbvf Intel(R) Gigabit Virtual Function Network Driver: 2.4.0-k
kernel module ioatdma : 4.00
kernel module ipmi_msghandler Incoming and outgoing message routing for an IPMI interface.: 39.2
kernel module ipr IBM Power RAID SCSI Adapter Driver: 2.6.4
kernel module ips IBM ServeRAID Adapter Driver 7.12.05: 7.12.05
kernel module iscsi : 1.2.0
kernel module ixgb Intel(R) PRO/10GbE Network Driver: 1.0.135-k2-NAPI
kernel module ixgbe Intel(R) 10 Gigabit PCI Express Network Driver: 5.1.0-k
kernel module ixgbe Intel(R) 10GbE PCI Express Linux Network Driver: 5.5.5
kernel module ixgbevf Intel(R) 10 Gigabit Virtual Function Network Driver: 4.1.0-k
kernel module jme JMicron JMC2x0 PCI Express Ethernet driver: 1.0.8
kernel module libcxgb Chelsio common library: 1.0.0-ko
kernel module libcxgbi Chelsio iSCSI driver library: 0.9.1-ko
kernel module liquidio Cavium LiquidIO Intelligent Server Adapter Driver: 1.7.2
kernel module liquidio_vf Cavium LiquidIO Intelligent Server Adapter Virtual Function Driver: 1.7.2
kernel module lpfc Emulex LightPulse Fibre Channel SCSI driver 12.0.0.6: 0
kernel module megaraid LSI Logic MegaRAID legacy driver: 2.00.4

kernel module megaraid_mbox LSI Logic MegaRAID Mailbox Driver: 2.20.5.1
kernel module megaraid_mm LSI Logic Management Module: 2.20.2.7
kernel module megaraid_sas Avago MegaRAID SAS Driver: 07.706.03.00-rc1
kernel module mlx4_core Mellanox ConnectX HCA low-level driver: 4.0-0
kernel module mlx4_en Mellanox ConnectX HCA Ethernet driver: 4.0-0
kernel module mlx5_core Mellanox 5th generation network adapters (ConnectX series) core driver: 5.0-0
kernel module mpt3sas LSI MPT Fusion SAS 3.0 Device Driver: 26.100.00.00
kernel module mptbase Fusion MPT base driver: 3.04.20
kernel module mptctl Fusion MPT misc device (ioctl) driver: 3.04.20
kernel module mptfc Fusion MPT FC Host driver: 3.04.20
kernel module mptsas Fusion MPT SAS Host driver: 3.04.20
kernel module mptscsih Fusion MPT SCSI Host driver: 3.04.20
kernel module mptspi Fusion MPT SPI Host driver: 3.04.20
kernel module mtip32xx Micron RealSSD PCIe Block Driver: 1.3.1
kernel module mvsas Marvell 88SE6440 SAS/SATA controller driver: 0.8.16
kernel module myri10ge Myricom 10G driver (10GbE): 1.5.3-1.534
kernel module netxen_nic QLogic/NetXen (1/10) GbE Intelligent Ethernet Driver: 4.0.82
kernel module nfp The Netronome Flow Processor (NFP) driver.: 4.19.84
kernel module nicpf Cavium Thunder NIC Physical Function Driver: 1.0
kernel module nicvf Cavium Thunder NIC Virtual Function Driver: 1.0
kernel module niu NIU ethernet driver: 1.1
kernel module nvme : 1.0
kernel module nvme_core : 1.0
kernel module pata_acpi SCSI low-level driver for ATA in ACPI mode: 0.2.3
kernel module pata_ali low-level driver for ALi PATA: 0.7.8
kernel module pata_amd low-level driver for AMD and Nvidia PATA IDE: 0.4.1
kernel module pata_artop SCSI low-level driver for ARTOP PATA: 0.4.6
kernel module pata_atiixp low-level driver for ATI IXP200/300/400: 0.4.6
kernel module pata_atp867x low level driver for Artop/Acard 867x ATA controller: 0.7.5
kernel module pata_cmd64x low-level driver for CMD64x series PATA controllers: 0.2.18
kernel module pata_efar SCSI low-level driver for EFAR PIIX clones: 0.4.5
kernel module pata_hpt366 low-level driver for the Highpoint HPT366/368: 0.6.11
kernel module pata_hpt37x low-level driver for the Highpoint HPT37x/30x: 0.6.23
kernel module pata_hpt3x2n low-level driver for the Highpoint HPT3xxN: 0.3.15
kernel module pata_hpt3x3 low-level driver for the Highpoint HPT343/363: 0.6.1
kernel module pata_it821x low-level driver for the IT8211/IT8212 IDE RAID controller: 0.4.2
kernel module pata_jmicron SCSI low-level driver for Jmicron PATA ports: 0.1.5
kernel module pata_marvell SCSI low-level driver for Marvell ATA in legacy mode: 0.1.6
kernel module pata_mpiix low-level driver for Intel MPIIX: 0.7.7
kernel module pata_netcell SCSI low-level driver for Netcell PATA RAID: 0.1.7
kernel module pata_ninja32 low-level driver for Ninja32 ATA: 0.1.5
kernel module pata_ns87410 low-level driver for Nat Semi 87410: 0.4.6
kernel module pata_ns87415 ATA low-level driver for NS87415 controllers: 0.0.1
kernel module pata_oldpiix SCSI low-level driver for early PIIX series controllers: 0.5.5
kernel module pata_pdc2027x libata driver module for Promise PDC20268 to PDC20277: 1.0
kernel module pata_pdc202xx_old low-level driver for Promise 2024x and 20262-20267: 0.4.3
kernel module pata_platform low-level driver for platform device ATA: 1.2
kernel module pata_rdc SCSI low-level driver for RDC PATA controllers: 0.01
kernel module pata_rz1000 low-level driver for RZ1000 PCI ATA: 0.2.4
kernel module pata_sch SCSI low-level driver for Intel SCH PATA controllers: 0.2
kernel module pata_serverworks low-level driver for Serverworks OSB4/CSB5/CSB6: 0.4.3
kernel module pata_sil680 low-level driver for SI680 PATA: 0.4.9
kernel module pata_sis SCSI low-level driver for SiS ATA: 0.5.2
kernel module pata_sl82c105 low-level driver for SI82c105: 0.3.3

kernel module pata_triflex low-level driver for Compaq Triflex: 0.2.8
 kernel module pata_via low-level driver for VIA PATA: 0.3.4
 kernel module pdc_adma Pacific Digital Corporation ADMA low-level driver: 1.0
 kernel module pm80xx PMC-Sierra PM8001/8006/8081/8088/8089/8074/8076/8077/8070/8072 SAS/SATA controller driver: 0.1.38
 kernel module pmcraid PMC Sierra MaxRAID Controller Driver: 1.0.3
 kernel module qed QLogic FastLinQ 4xxxx Core Module: 8.33.0.20
 kernel module qede QLogic FastLinQ 4xxxx Ethernet Driver: 8.33.0.20
 kernel module qedf QLogic QEDF 25/40/50/100Gb FCoE Driver: 8.33.16.20
 kernel module qedi QLogic FastLinQ 4xxxx iSCSI Module: 8.33.0.20
 kernel module qla1280 Qlogic ISP SCSI (qla1x80/qla1x160) driver: 3.27.1
 kernel module qla2xxx QLogic Fibre Channel HBA Driver: 10.00.00.08-k
 kernel module qla3xxx QLogic ISP3XXX Network Driver v2.03.00-k5 : v2.03.00-k5
 kernel module qla4xxx QLogic iSCSI HBA Driver: 5.04.00-k6
 kernel module qlcnic QLogic 1/10 GbE Converged/Intelligent Ethernet Driver: 5.3.66
 kernel module qlge QLogic 10 Gigabit PCI-E Ethernet Driver : 1.00.00.35
 kernel module r6040 RDC R6040 NAPI PCI FastEthernet driver: 0.29 04Jul2016
 kernel module rsxx IBM Flash Adapter 900GB Full Height Device Driver: 4.0.3.2516
 kernel module s2io : 2.0.26.28
 kernel module sata_dwc_460ex DesignWare Cores SATA controller low level driver: 1.3
 kernel module sata_mv SCSI low-level driver for Marvell SATA controllers: 1.28
 kernel module sata_nv low-level driver for NVIDIA nForce SATA controller: 3.5
 kernel module sata_promise Promise ATA TX2/TX4/TX4000 low-level driver: 2.12
 kernel module sata_qstor Pacific Digital Corporation QStor SATA low-level driver: 0.09
 kernel module sata_sil low-level driver for Silicon Image SATA controller: 2.4
 kernel module sata_sis low-level driver for Silicon Integrated Systems SATA controller: 1.0
 kernel module sata_svw low-level driver for K2 SATA controller: 2.3
 kernel module sata_sx4 Promise SATA low-level driver: 0.12
 kernel module sata_uli low-level driver for ULi Electronics SATA controller: 1.3
 kernel module sata_via SCSI low-level driver for VIA SATA controllers: 2.6
 kernel module sata_vsc low-level driver for Vitesse VSC7174 SATA controller: 2.3
 kernel module sfc Solarflare network driver: 4.1
 kernel module sfc_falcon Solarflare Falcon network driver: 4.1
 kernel module sg SCSI generic (sg) driver: 3.5.36
 kernel module sis190 SiS sis190/191 Gigabit Ethernet driver: 1.4
 kernel module skge SysKonnect Gigabit Ethernet driver: 1.14
 kernel module sky2 Marvell Yukon 2 Gigabit Ethernet driver: 1.30
 kernel module slicoss Alacritech non-accelerated SLIC driver: 1.0
 kernel module smartpqi Driver for Microsemi Smart Family Controller version 1.1.4-130: 1.1.4-130
 kernel module smsc911x : 2008-10-21
 kernel module smsc9420 : 1.01
 kernel module snic Cisco SCSI NIC Driver: 0.0.1.18
 kernel module starfire Adaptec Starfire Ethernet driver: 2.1
 kernel module stex Promise Technology SuperTrak EX Controllers: 6.02.0000.01
 kernel module sunhme Sun HappyMealEthernet(HME) 10/100baseT ethernet driver: 3.10
 kernel module sym53c8xx NCR, Symbios and LSI 8xx and 1010 PCI SCSI adapters: 2.2.3
 kernel module tg3 Broadcom Tigon3 ethernet driver: 3.137
 kernel module thunder_bgx Cavium Thunder BGX/MAC Driver: 1.0
 kernel module thunder_xcv Cavium Thunder RGX/XCV Driver: 1.0
 kernel module tpm TPM Driver: 2.0
 kernel module tpm_atmel TPM Driver: 2.0
 kernel module tpm_crb TPM2 Driver: 0.1
 kernel module tpm_i2c_infineon TPM TIS I2C Infineon Driver: 2.2.0
 kernel module tpm_infineon Driver for Infineon TPM SLD 9630 TT 1.1 / SLB 9635 TT 1.2: 1.9.2
 kernel module tpm_nsc TPM Driver: 2.0

kernel module tpm_st33zp24 ST33ZP24 TPM 1.2 driver: 1.3.0
kernel module tpm_st33zp24_i2c STM TPM 1.2 I2C ST33 Driver: 1.3.0
kernel module tpm_tis TPM Driver: 2.0
kernel module tpm_tis_core TPM Driver: 2.0
kernel module tpm_vtpm_proxy vTPM Driver: 0.1
kernel module tulip Digital 21*4* Tulip ethernet driver: 1.1.15
kernel module typhoon 3Com Typhoon Family (3C990, 3CR990, and variants): 1.0
kernel module ufshcd_core Generic UFS host controller driver Core: 0.2
kernel module virtio_pci virtio-pci: 1
kernel module vmw_pvscsi VMware PVSCSI driver: 1.0.7.0-k
kernel module vmxnet3 VMware vmxnet3 virtual NIC driver: 1.4.16.0-k
kernel module vxlan Driver for VXLAN encapsulated traffic: 0.1
kernel module winbond_840 Winbond W89c840 Ethernet driver: 1.01-e
adduser version: 3.118
apt version: 1.8.2
apt-utils version: 1.8.2
base-files version: 10.3+deb10u3
base-passwd version: 3.5.46
bash version: 5.0-4
bsdmainutils version: 11.1.2+b1
bsdutils version: 1:2.33.1-0.1
busybox version: 1:1.30.1-4
bzip2 version: 1.0.6-9.2~deb10u1
ca-certificates version: 20190110
coreutils version: 8.30-3
cpio version: 2.12+dfsg-9
cron version: 3.0pl1-134+deb10u1
cryptsetup-bin version: 2:2.1.0-5+deb10u2
curl version: 7.64.0-4
dash version: 0.5.10.2-5
dbus version: 1.12.16-1
debconf version: 1.5.71
debconf-i18n version: 1.5.71
debian-archive-keyring version: 2019.1
debiantools version: 4.8.6.1
diffutils version: 1:3.7-3
dirmngr version: 2.2.12-1+deb10u1
dmidecode version: 3.2-1
dmsetup version: 2:1.02.155-3
dosfstools version: 4.1-2
dpkg version: 1.19.7
e2fsprogs version: 1.44.5-1+deb10u3
ethtool version: 1:4.19-1
fdisk version: 2.33.1-0.1
file version: 1:5.35-4+deb10u1
findutils version: 4.6.0+git+20190209-2
gcc-8-base version: 8.3.0-6
gdbm-l10n version: 1.18.1-4
gdisk version: 1.0.3-1.1
gnupg version: 2.2.12-1+deb10u1
gnupg-l10n version: 2.2.12-1+deb10u1
gnupg-utils version: 2.2.12-1+deb10u1
gpg version: 2.2.12-1+deb10u1
gpg-agent version: 2.2.12-1+deb10u1

gpg-wks-client version: 2.2.12-1+deb10u1
gpg-wks-server version: 2.2.12-1+deb10u1
gpgconf version: 2.2.12-1+deb10u1
gpgsm version: 2.2.12-1+deb10u1
gpgv version: 2.2.12-1+deb10u1
grep version: 3.3-1
groff-base version: 1.22.4-3
guile-2.2-libs version: 2.2.4+1-2+deb10u1
gzip version: 1.9-3
hdparm version: 9.58+ds-1
hostname version: 3.21
hwdm version: 21.63-3
ifenslave version: 2.9
ifenslave-2.6 version: 2.9
ifupdown version: 0.8.35
init version: 1.56+nmu1
init-system-helpers version: 1.56+nmu1
initramfs-tools version: 0.133+deb10u1
initramfs-tools-core version: 0.133+deb10u1
iproute2 version: 4.20.0-2
iptables version: 1.8.2-4
iputils-ping version: 3:20180629-2
irqbalance version: 1.5.0-3
isc-dhcp-client version: 4.4.1-2
isc-dhcp-common version: 4.4.1-2
klibc-utils version: 2.0.6-1
kmod version: 26-1
kpartx version: 0.7.9-3
krb5-locales version: 1.17-3
less version: 487-0.1+b1
libacl1 version: 2.2.53-4
libaio1 version: 0.3.112-3
libapparmor1 version: 2.13.2-10
libapt-inst2.0 version: 1.8.2
libapt-pkg5.0 version: 1.8.2
libargon2-1 version: 0~20171227-0.2
libassuan0 version: 2.5.2-1
libattr1 version: 1:2.4.48-4
libaudit-common version: 1:2.8.4-3
libaudit1 version: 1:2.8.4-3
libblkid1 version: 2.33.1-0.1
libboost-atomic1.67.0 version: 1.67.0-13+deb10u1
libboost-python1.67.0 version: 1.67.0-13+deb10u1
libboost-system1.67.0 version: 1.67.0-13+deb10u1
libboost-thread1.67.0 version: 1.67.0-13+deb10u1
libbsd0 version: 0.9.1-2
libbz2-1.0 version: 1.0.6-9.2~deb10u1
libc-bin version: 2.28-10
libc6 version: 2.28-10
libcap-ng0 version: 0.7.9-2
libcap2 version: 1:2.25-2
libcap2-bin version: 1:2.25-2
libcom-err2 version: 1.44.5-1+deb10u3
libcryptsetup12 version: 2:2.1.0-5+deb10u2

libcurl4 version: 7.64.0-4
libdb5.3 version: 5.3.28+dfsg1-0.5
libdbus-1-3 version: 1.12.16-1
libdebconfclient0 version: 0.249
libdevmapper1.02.1 version: 2:1.02.155-3
libdns-export1104 version: 1:9.11.5.P4+dfsg-5.1
libedit2 version: 3.1-20181209-1
libelf1 version: 0.176-1.1
libestr0 version: 0.1.10-2.1
libevent-core-2.1-6 version: 2.1.8-stable-4
libevent-pthreads-2.1-6 version: 2.1.8-stable-4
libexpat1 version: 2.2.6-2+deb10u1
libext2fs2 version: 1.44.5-1+deb10u3
libfastjson4 version: 0.99.8-2
libfdisk1 version: 2.33.1-0.1
libffi6 version: 3.2.1-9
libfribidi0 version: 1.0.5-3.1+deb10u1
libgc1c2 version: 1:7.6.4-0.4
libgcc1 version: 1:8.3.0-6
libgcrypt20 version: 1.8.4-5
libgdbm-compat4 version: 1.18.1-4
libgdbm6 version: 1.18.1-4
libglib2.0-0 version: 2.58.3-2+deb10u2
libglib2.0-data version: 2.58.3-2+deb10u2
libgmp10 version: 2:6.1.2+dfsg-4
libgnutls-openssl27 version: 3.6.7-4+deb10u2
libgnutls30 version: 3.6.7-4+deb10u2
libgpg-error0 version: 1.35-1
libgsasl7 version: 1.8.0-8+b2
libgssapi-krb5-2 version: 1.17-3
libhd21 version: 21.63-3
libhogweed4 version: 3.4.1-1
libicu63 version: 63.1-6
libidn11 version: 1.33-2.2
libidn2-0 version: 2.0.5-1+deb10u1
libip4tc0 version: 1.8.2-4
libip6tc0 version: 1.8.2-4
libiptc0 version: 1.8.2-4
libisc-export1100 version: 1:9.11.5.P4+dfsg-5.1
libjson-c3 version: 0.12.1+ds-2
libk5crypto3 version: 1.17-3
libkeyutils1 version: 1.6-6
libklibc version: 2.0.6-1
libkmod2 version: 26-1
libkrb5-3 version: 1.17-3
libkrb5support0 version: 1.17-3
libksba8 version: 1.3.5-2
libkyotocabinet16v5 version: 1.2.76-4.2+b1
libldap-2.4-2 version: 2.4.47+dfsg-3+deb10u1
libldap-common version: 2.4.47+dfsg-3+deb10u1
liblocale-gettext-perl version: 1.07-3+b4
liblognorm5 version: 2.0.5-1
libltdl7 version: 2.4.6-9
liblz4-1 version: 1.8.3-1

liblzma5 version: 5.2.4-1
liblzo2-2 version: 2.10-0.1
libmagic-mgc version: 1:5.35-4+deb10u1
libmagic1 version: 1:5.35-4+deb10u1
libmailutils5 version: 1:3.5-3
libmariadb3 version: 1:10.3.22-0+deb10u1
libmnl0 version: 1.0.4-2
libmount1 version: 2.33.1-0.1
libmpdec2 version: 2.4.2-2
libncurses6 version: 6.1+20181013-2+deb10u2
libncursesw6 version: 6.1+20181013-2+deb10u2
libnetfilter-contrack3 version: 1.0.7-1
libnettle6 version: 3.4.1-1
libnewt0.52 version: 0.52.20-8
libnfnetwork0 version: 1.0.1-3+b1
libnftnl11 version: 1.1.2-2
libnghttp2-14 version: 1.36.0-2+deb10u1
libnpt0 version: 1.6-1
libntlm0 version: 1.5-1
libnuma1 version: 2.0.12-1
libopenipmi0 version: 2.0.25-2.1
libopts25 version: 1:5.18.12-4
libp11-kit0 version: 0.23.15-2
libpam-modules version: 1.3.1-5
libpam-modules-bin version: 1.3.1-5
libpam-runtime version: 1.3.1-5
libpam-systemd version: 241-7~deb10u3
libpam0g version: 1.3.1-5
libpcap0.8 version: 1.8.1-6
libpci3 version: 1:3.5.2-1
libpcre3 version: 2:8.39-12
libperl5.28 version: 5.28.1-6
libpopt0 version: 1.16-12
libprocps7 version: 2:3.3.15-2
libpsl5 version: 0.20.2-2
libpython2.7 version: 2.7.16-2+deb10u1
libpython2.7-minimal version: 2.7.16-2+deb10u1
libpython2.7-stdlib version: 2.7.16-2+deb10u1
libpython3.7 version: 3.7.3-2+deb10u1
libpython3.7-minimal version: 3.7.3-2+deb10u1
libpython3.7-stdlib version: 3.7.3-2+deb10u1
libreadline7 version: 7.0-5
librtmp1 version: 2.4+20151223.gitfa8646d.1-2
libsasl2-2 version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules version: 2.1.27+dfsg-1+deb10u1
libsasl2-modules-db version: 2.1.27+dfsg-1+deb10u1
libseccomp2 version: 2.3.3-4
libselinux1 version: 2.8-1+b1
libsemanage-common version: 2.8-2
libsemanage1 version: 2.8-2
libsensors-config version: 1:3.5.0-3
libsensors5 version: 1:3.5.0-3
libsepol1 version: 2.8-1
libsgutils2-2 version: 1.44-1

libslang2 version: 2.3.2-2
libsmartcols1 version: 2.33.1-0.1
libsnmp-base version: 5.7.3+dfsg-5
libsnmp30 version: 5.7.3+dfsg-5
libsqlite3-0 version: 3.27.2-3
libss2 version: 1.44.5-1+deb10u3
libssh2-1 version: 1.8.0-2.1
libssl1.1 version: 1.1.1d-0+deb10u2
libstdc++6 version: 8.3.0-6
libsysfs2 version: 2.1.0+repack-5
libsystemd0 version: 241-7~deb10u3
libtasn1-6 version: 4.13-3
libtext-charwidth-perl version: 0.04-7.1+b1
libtext-iconv-perl version: 1.7-5+b7
libtext-wrapi18n-perl version: 0.06-7.1
libtinfo6 version: 6.1+20181013-2+deb10u2
libuchardet0 version: 0.0.6-3
libudev1 version: 241-7~deb10u3
libunistring2 version: 0.9.10-1
liburcu6 version: 0.10.2-1
libuuid1 version: 2.33.1-0.1
libwrap0 version: 7.6.q-28
libx11-6 version: 2:1.6.7-1
libx11-data version: 2:1.6.7-1
libx86emu2 version: 2.0-1
libxau6 version: 1:1.0.8-1+b2
libxcb1 version: 1.13.1-2
libxdmcp6 version: 1:1.1.2-3
libxext6 version: 2:1.3.3-1+b2
libxml2 version: 2.9.4+dfsg1-7+b3
libxmuu1 version: 2:1.1.2-2+b3
libxtables12 version: 1.8.2-4
libyajl2 version: 2.1.0-3
libzstd1 version: 1.3.8+dfsg-3
linux-base version: 4.6
linux-firmware version: 1.183.2
login version: 1:4.5-1.1
logrotate version: 3.14.0-4
lsb-base version: 10.2019051400
lsscsi version: 0.30-0.1
mailutils version: 1:3.5-3
mailutils-common version: 1:3.5-3
mariadb-common version: 1:10.3.22-0+deb10u1
mawk version: 1.3.3-17+b3
mime-support version: 3.62
mount version: 2.33.1-0.1
multipath-tools version: 0.7.9-3
multipath-tools-boot version: 0.7.9-3
mysql-common version: 5.8+1.0.5
nano version: 3.2-3
ncurses-base version: 6.1+20181013-2+deb10u2
ncurses-bin version: 6.1+20181013-2+deb10u2
ncurses-term version: 6.1+20181013-2+deb10u2
net-tools version: 1.60+git20180626.aebd88e-1

netbase version: 5.6
nload version: 0.7.4-2+b1
ntp version: 1:4.2.8p12+dfsg-4
numactl version: 2.0.12-1
openipmi version: 2.0.25-2.1
openssh-client version: 1:7.9p1-10+deb10u2
openssh-server version: 1:7.9p1-10+deb10u2
openssh-sftp-server version: 1:7.9p1-10+deb10u2
openssl version: 1.1.1d-0+deb10u2
passwd version: 1:4.5-1.1
perl version: 5.28.1-6
perl-base version: 5.28.1-6
perl-modules-5.28 version: 5.28.1-6
pigz version: 2.4-1
pinentry-curses version: 1.1.0-2
powermgmt-base version: 1.34
procps version: 2:3.3.15-2
publicsuffix version: 20190415.1030-1
python3.7 version: 3.7.3-2+deb10u1
python3.7-minimal version: 3.7.3-2+deb10u1
qemu-guest-agent version: 1:3.1+dfsg-8+deb10u3
readline-common version: 7.0-5
rsyslog version: 8.1901.0-1
runit-helper version: 2.8.6
sdparm version: 1.10-1
sed version: 4.7-1
sensible-utils version: 0.0.12
sg3-utils version: 1.44-1
sg3-utils-udev version: 1.44-1
shared-mime-info version: 1.10-1
smartmontools version: 6.6-1
smp-utils version: 0.98-2
snmp version: 5.7.3+dfsg-5
snmpd version: 5.7.3+dfsg-5
snmp version: 1:4.2.8p12+dfsg-4
ssmtp version: 2.64-8.1
sysstat version: 12.0.3-2
systemd version: 241-7~deb10u3
systemd-sysv version: 241-7~deb10u3
sysvinit-utils version: 2.93-8
tar version: 1.30+dfsg-6
tasksel version: 3.53
tasksel-data version: 3.53
tcpdump version: 4.9.3-1~deb10u1
tofrodos version: 1.7.13+ds-4
traceroute version: 1:2.1.0-2
tzdata version: 2019c-0+deb10u1
ucf version: 3.0038+nmu1
udev version: 241-7~deb10u3
util-linux version: 2.33.1-0.1
vim-common version: 2:8.1.0875-5
vim-tiny version: 2:8.1.0875-5
whiptail version: 0.52.20-8
xauth version: 1:1.0.10-1

xdg-user-dirs version: 0.17-2
xxd version: 2:8.1.0875-5
xz-utils version: 5.2.4-1
zlib1g version: 1:1.2.11.dfsg-1
linux-firmware: 1.183.2
ssmtp: 2.64-8.1

Elasticsearch-specific and additional Caringo distributions:

Elasticsearch 6.8.6 / 5.6.12 / 2.3.3
elasticsearch-curator 4.3.1 (supports Elasticsearch 5 and 6)
txes 0.1.4+
Swarm S3 Backup Restore 1.1.0
Swarm Search 6.0.2
Swarm Metrics 6.0.2

Swarm Storage 11.0 Release

New Features

S3 Backup and Restore – In addition to on-premises Swarm storage and remote clusters, you can now take advantage of public cloud services for off-premises disaster recovery (DR) storage. Amazon S3 has the widest support in the industry, and Swarm Content Gateway already supports S3, so S3 is the first cloud destination from Swarm. By implementing an S3 backup feed from Swarm, you have the security of knowing backups are continuous, have minimal latency, and require little intervention and monitoring by you. Using Swarm's feed mechanism for backup leverages numerous existing strengths: its long-term iteration over objects in the cluster, proven method for tracking work as it is performed, and mechanisms for TLS connections and forward proxies. Having the parallelism of the entire cluster makes best use of your network bandwidth, while sending the backups through a forward proxy enables bandwidth throttling.

- **Back up** – S3 Backup occurs as an integral part of your operating Swarm cluster. In the Swarm UI, create a new feed of type S3 Backup, provide credentials and information about the network path to the service. After the feed is started, you can monitor its progress and be warned of blockages and particular object failures, as with any other feed. The S3 Backup feed honors the versioning settings in your cluster, as enabled, disabled, or suspended throughout the domains and buckets. See [S3 Backup Feeds](#).
- **Clean up** – No action on your part is needed to keep the backup current and trimmed. When versioning is disabled in Swarm on buckets or domains, delete buckets or domains, or have object lifepoints expire, the Swarm feeds mechanism processes the expired content as deleted, allowing the S3 Backup feed to clear them from the S3 bucket. Throughout content additions and deletions, the total number of objects in an S3 bucket is approximately twice the number of logical objects backing up from the source cluster.
- **Restore** – The Restore tool runs outside of Swarm, using a command-line interface for executing the data and restoration tasks. You can restore what you need: either the entire cluster, or only portions. Swarm supports bulk restores at the granularity of cluster, domain, or bucket, as well as more surgical restores of a few objects. You can also run multiple copies to achieve a faster, parallel recovery. See the [S3 Backup Restore Tool](#).

Faster Volume Mounting – Due to re-engineering of disk mounting and common disk operations, Swarm 11 has a 30% improvement in volume mount times over previous versions. (SWAR-7957)

Prometheus Node Exporter – To make your Prometheus node exporter metrics named for global uniqueness and also ease of identification, Swarm now prefixes the Prometheus node exporter metrics with 'caringo_swarm_' instead of 'metrics_'. (SWAR-8539) In addition, the setting `metrics.nodeExporterFrequency` is now a persisted cluster setting with MIB name `metricsExporterFrequency`. See [Prometheus Node Exporter and Grafana](#). (SWAR-8467)

System Status on Console – On the System Menu accessed from the physical console of a Swarm node, the Diagnostics Menu has additional functionality for viewing system status. The new options include *Systemd Unit Status*, *Systemd journal*, and *Top processes* list. (SWAR-3412)

Improved Memory Management – Swarm 11 includes changes for better memory management in low memory situations. In Swarm 10, insufficient memory on a node for all volumes being managed causes Swarm to reboot; with these improvements, rebooting is less likely. Verify each node meets a minimum physical memory of 2 GB + (0.5 GB * number of volumes) for best results. More memory benefits Swarm's performance. (SWAR-8558)

Container-Compatible – The architecture work of Swarm 10 continues with build-out of support for containerization, so Swarm storage nodes can now be managed in containers.

Large Cluster Performance – This release includes performance improvements for very large clusters, which benefits clusters of all sizes. (11.0.1: SWAR-8616)

Additional Changes

These items are other changes and improvements including those that come from testing and user feedback.

- **OSS Versions** – See [Third-Party Components for 11.0](#) for the complete listing of packages and versions.
 - The Linux kernel is upgraded to 4.19.56, which mitigates Linux Sack vulnerability. (SWAR-8534)
 - Linux firmware is upgraded to 1.179. (SWAR-8341)
 - Numerous network drivers are updated, including bnx2, bnx2x, ixgbe, and i40; see the [complete listing](#) for variants and versions. (SWAR-8341)
- **Fixed in 11.0.3**
 - A kernel configuration issue prevented the discovery of ATA disks attached to an SAS controller. (SWAR-8663)
- **Fixed in 11.0.2**
 - *Improved:* When Swarm completes a retire task, the announce-level message it generates now reports the overall duration and rate of the retire. (SWAR-8633)
 - The health processor does not always clear memory of replicas on long-removed volumes, which caused periodic FVRs. (SWAR-8639)
 - Swarm 11.0.0 showed an incorrect value (11.0.0.rc8) for its build revision. (SWAR-8627)
 - When recoveries of specific volumes are suspended by SNMP or API calls, those recoveries still appear to be running. (SWAR-8604)
 - The health processor state (healthProcessorState in SNMP) sometimes showed "idle" when health processing was paused for failed volume recoveries (FVRs). (SWAR-8601)
 - Retiring volumes are reported as available space even though they cannot be written to. (SWAR-7865)
 - Under some conditions, Swarm may start without mounting some of its volumes. (SWAR-8597)
- **Fixed in 11.0.0**
 - The node console's system menu can be obscured by stray text from the boot process. (SWAR-8591)
 - A dmesg dump (on the Chassis Details page or the legacy Admin Console) may be missing some or all driver messages. (SWAR-8573)
 - Although the bucket existed, erroneous CRITICAL messages may report that "Bucket (uuid=...) in domain '...' has been deleted with orphan content." (SWAR-8560)
 - Too many replicas of context objects (buckets and domains) caused error messages about being unable to index objects. After upgrading, these messages stop once several HP cycles are able to complete. (SWAR-8555)
 - The OS in 10.2.1 cannot mount USB flash drives and so cannot read node.cfg files from them. (SWAR-8501)
 - Swarm now prevents and removes any overage caused by erroneous remote replication of EC streams via a replication feed, which can double the space usage. (SWAR-8439)
 - While a node is down for maintenance, erroneous CRITICAL errors may report that EC objects have insufficient protection. (SWAR-8421)
 - Swarm returns a 410 *Gone* response (instead of 412 *Precondition Failed*) for unrecoverable multipart upload requests. (SWAR-8343)
 - On getting new capacity, fuller clusters are slow to rebalance over the available volumes. (SWAR-8116)

Upgrade Impacts

These items are changes to the product function that may require operational or development changes for integrated applications. Address the upgrade impacts for each of the versions since the one you are currently running:

Impacts for 11.0

- **Upgrading Elasticsearch** – You may use Elasticsearch 2.3.3 with Storage 11.0 if you cannot move to 5.6 now, but plan the migration immediately (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release, and testing with Swarm 11 is discontinuing.

- **Propagate Deletes Deprecated** – The option to disable *Propagate Deletes* on [replication feeds](#) is deprecated; use [Object Versioning](#) to preserve deleted content. Do not disable Propagate Deletes when versioning is enabled or when defining an S3 Backup. (SWAR-8609)
- **Configuration Settings** – Run the [Storage Settings Checker](#) to identify these and other configuration issues.
 - Changed settings:
 - `ec.segmentConsolidationFrequency` (`ecSegmentConsolidationFrequency` in SNMP) has an improved default (10), which you must apply to your cluster when you upgrade. (SWAR-8483)
 - `cluster.name` is now required. Add it to the `cluster.cfg` file. (SWAR-8466).
 - `metrics.nodeExporterFrequency` (`metricsExporterFrequency` in SNMP) is now a persisted cluster setting. (SWAR-8467).
 - Removed settings:
 - `chassis.processes` is allowed but is ignored.
 - Numerous settings are now promoted to *cluster-level* (versus node-level) scope, so you can manage them via **Settings > Cluster** in the Swarm UI (SWAR-8457):
 - `console.expiryErrInterval`
 - `console.expiryWarnInterval`
 - `console.indexErrorLevel`
 - `console.indexWarningLevel`
 - `console.port`
 - `console.reportStyleUrl`
 - `console.spaceErrorLevel`
 - `console.spaceWarnLevel`
 - `console.styleUrl`
 - `feeds.retry`
 - `feeds.statsReportInterval`
 - `health.parallelWriteTimeout`
 - `log.obscureUUIDs`
 - `metrics.enableNodeExporter`
 - `network.dnsDomain`
 - `network.dnsServers`
 - `network.icmpAcceptRedirects`
 - `network.igmpVersion`
 - `network.mtu`
 - `startup.certificates`

Impacts for 10.2

- **Upgrading Elasticsearch** – You may continue to use Elasticsearch 2.3.3 with Storage 10.2 until you are able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release. the upgrade to Elasticsearch 5.6 must be completed before [upgrading to Gateway 6.0](#).
- **Configuration Settings** – Run the [Storage Settings Checker](#) before any Swarm 10 upgrade to identify configuration issues. Note these changes:
 - `ec.protectionLevel` is now persisted. (SWAR-8231)
 - `index.ovMinNodes=3` is the new default for the overlay index, in support of Swarm 10's new architecture. To keep your overlay index operational, set this new value in your cluster, through the UI or by SNMP (`overlayMinNodes`). (SWAR-8278)

- `metrics.enableNodeExporter` can be set to True, which enables the Prometheus Node Exporter on that node. (SWAR-8408, SWAR-8578)
- `metrics.nodeExporterFrequency`, a new dynamic setting, sets how frequently to refresh Swarm-specific Prometheus metrics in Elasticsearch; it defaults to 0, which disables this export. (SWAR-8408).

Impacts for 10.1

- **Upgrading Elasticsearch** – Continue to use Elasticsearch 2.3.3 with Storage 10.1 until able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release. Complete the upgrade to Elasticsearch 5.6 before upgrading to Gateway 6.0.
- **Configuration Settings** – Run the [Storage Settings Checker](#) before any Swarm 10 upgrade to identify configuration issues.
 - `metrics.enableNodeExporter=true` enables Swarm to run the Prometheus node exporter on port 9100. (SWAR-8170)
- **IP address update delay** – When upgrading from Swarm 9 to the new architecture of Swarm 10, note the "ghosts" of previously used IP addresses may appear in the Storage UI; these resolve within 4 days. (SWAR-8351)
- **Update MIBs on CSN** – Before upgrading to Storage 10.x, the MIBs on the CSN must be updated. From the Swarm Support tools bundle, run the `platform-update-mibs.sh` script. (CSN-1872)

Impacts for 10.0

- **Upgrading Elasticsearch** – You may continue to use Elasticsearch 2.3.3 with Storage 10.0 until you are able to move to 5.6 (see [Migrating from Older Elasticsearch](#)). Support for ES 2.3.3 ends in a future release.
- **Configuration Settings** – Run the [Storage Settings Checker](#) to identify these and other configuration issues.
 - Changes for the new single-IP dense architecture:
 - `network.ipAddress` – multiple IP addresses now disallowed
 - `chassis.processes` – removed; multi-server configurations are no longer supported
 - `ec.protectionLevel` – new value "volume"
 - `ec.subclusterLossTolerance` – removed
 - Changes for security (see next section)
 - `security.administrators`, `security.operators` – removed 'snmp' user
 - `snmp.rwCommunity`, `snmp.roCommunity` – new settings for 'snmp' user
 - `startup.certificates` – new setting to hold any and all public keys
 - New settings:
 - `disk.atimeEnabled`
 - `health.parallelWriteTimeout`
 - `search.pathDelimiter`
- **Required SNMP security change** – Remove the `snmp` key from the `security.administrators` setting, and update `snmp.rwCommunity` with its value. Nodes that contain only the `snmp` key in the `security.administrators` setting does not boot. If you changed the default value of the `snmp` key in the `security.operators` setting, update `snmp.roCommunity` with that value and then remove the `snmp` key from `security.operators`. In the `security.operators` setting, 'snmp' is a reserved key, and it cannot be an authorized console operator name. (SWAR-8097)
- **EC protection**
 - *Best practice:* Use `ec.protectionLevel=node`, which distributes segments across the cluster's physical/virtual machines. Do not use `ec.protectionLevel=subcluster` unless you already have subclusters defined and are sure the specified EC encoding is supported. A new level, `ec.protectionLevel=volume`, allows EC writes to succeed if you have a small cluster with fewer than $(k+p)/p$ nodes. (Swarm always seeks the highest protection possible for EC segments, regardless of the level you set.)

- Optimize hardware for EC by verifying there are more than $k+p$ subclusters/nodes (as set by `ec.protectionLevel`); for example, with `policy.ecEncoding=5:2`, you need at least 8 subclusters/nodes. When Swarm cannot distribute EC segments adequately for protection, EC writes can fail despite ample free space. (SWAR-7985)
- Setting `ec.protectionLevel=subcluster` without creating subclusters (defining `node.subcluster` across sets of nodes) causes a critical error and lowers the protection level to 'node'. (SWAR-8175)
- **Small clusters** – Verify the following settings if using 10 or fewer Swarm nodes. Do not use fewer than 3 in production. *Important:* If you need to change any, do so *before* upgrading to Swarm 10.
 - **policy.replicas** – The `min` and `default` values for numbers of replicas to keep in your cluster must not exceed your number of nodes. For example, a 3-node cluster may have only `min=2` or `min=3`.
 - **EC encoding and protection** – For EC encoding, verify you have enough nodes to support the cluster's encoding (`policy.ecEncoding`). For EC writes to succeed with fewer than $(k+p)/p$ nodes, use the new level, `ec.protectionLevel=volume`.
 - *Best practice:* Keep at least one physical machine in your cluster beyond the minimum number needed. This allows for one machine to be down for maintenance without compromising the constraint.
- **"Cluster in a box"** – Swarm supports a "cluster in a box" configuration as long as that box is running a virtual machine host and Swarm instances are running in 3 or more VMs. Each VM boots separately and has its own IP address. Follow the recommendations for small clusters, substituting VMs for nodes. If you have two physical machines, use the "cluster in a box" configuration, but move to direct booting of Swarm with 3 or more.
- **Offline node status** – Because Swarm 10's new architecture reduces the number of IP addresses in your storage cluster, you may see the old IPs and subclusters reporting as **Offline** nodes until they timeout in 4 days (`crier.forgetOfflineInterval`), which is expected.

For Swarm 9 impacts, see [Swarm Storage 9 Releases](#).

Watch Items and Known Issues

The following operational limitations and watch items exist in this release.

- When using ES 5.6, deprecation warnings can cause logs to consume excessive disk space. Workaround: To exclude the warnings, add 'logger.deprecation.level = error' to the top of the `log4j2.properties` file. (SWAR-8632)
- Swarm 11.0.0 shows an incorrect value (11.0.0.rc8) for its build revision. (SWAR-8627)
- Under some conditions, Swarm may start without mounting some of its volumes. If this happens, reboot the node. (SWAR-8597)
- S3 Backup feeds do not back up logical objects greater than 5 GB. (SWAR-8554)
- If you downgrade from Swarm 11.0, CRITICAL errors may appear on your feeds. To stop the errors, edit the existing feed definition names via the Swarm UI or legacy Admin Console. (SWAR-8543)
- When restarting a cluster of virtual machines that are UEFI-booted (versus legacy BIOS), the chassis shut down but do not come back up. (SWAR-8054)
- If the Elasticsearch cluster is wiped, the Storage UI shows no NFS config. Contact DataCore Support for help repopulating your SwarmFS config information. (SWAR-8007)
- If a bucket is deleted, any incomplete multipart upload into that bucket leaves the parts (unnamed streams) in the domain. To find and delete them, use the `s3cmd` utility (search the Support site for "s3cmd" for guidance). (SWAR-7690)
- Logs showed the error "FEEDS WARNING: calcFeedInfo(etag=xxx) cannot find domain xxx, which is needed for a domains-specific replication feed". The root cause is fixed; if you received such warnings, contact DataCore Support so the issue can be resolved. (SWAR-7556)
- With multipath-enabled hardware, the Swarm console Disk Volume Menu may erroneously show too many disks, having multiplied the actual disks in use by the number of possible paths to them. (SWAR-7248)

Note these installation issues:

- The `elasticsearch-curator` package may show an error during an upgrade, which is a known curator issue. Workaround: Reinstall the curator: `yum reinstall elasticsearch-curator` (SWAR-7439)

- Do not install the Swarm Search RPM before installing Java. If Gateway startup fails with "Caringo script plugin is missing from indexer nodes", uninstall and reinstall the Swarm Search RPM. (SWAR-7688)

Upgrading Swarm

Proceed to [How to Upgrade Swarm](#) to upgrade Swarm 9 or higher.



Important

Contact DataCore Support for guidance if needing to upgrade from Swarm 8.x or earlier.

Third-Party Components for 11.0

zope.interface version: 4.6.0
ipaddress version: 1.0.22
cryptography version: 2.6.1
pyOpenSSL version: 19.0.0
service_identity version: 18.1.0
incremental version: 17.5.0
Twisted[tls] version: 18.9.0
egenix-mx-base version: 3.2.9
zbase32 version: 1.1.5
pyutil version: 3.1.0
python-dateutil version: 2.8.0
guppy version: 0.1.10
Werkzeug version: 0.15.2
klein version: 17.10.0
requests version: 2.21.0
certifi version: *latest as of 2019-07-25*
zfec-1.4.22.tar.gz:
CAStor SDK version: 6.1.5.1-py2.5
Yajl version: 2.1.0-0-ga0ecdde
Newt version: 0.52.20
megactl version: 0.4.1
Prometheus node_exporter version: 0.17.0-rc.0
libpng version: 1.2.8
LILO version: 22.7.1
Mock library version: 1.0.1
treq version: 0.2.0
pstat.py version: 0.4
Operating system: Debian GNU/Linux 9 (stretch)
Linux kernel: 4.19.56
kernel module 3w_9xxx 3ware 9000 Storage Controller Linux Driver: 2.26.02.014
kernel module 3w_sas LSI 3ware SAS/SATA-RAID Linux Driver: 3.26.02.000
kernel module 3w_xxxx 3ware Storage Controller Linux Driver: 1.26.02.003
kernel module 8021q : 1.8
kernel module 8139cp RealTek RTL-8139C+ series 10/100 PCI Ethernet driver: 1.3
kernel module 8139too RealTek RTL-8139 Fast Ethernet driver: 0.9.28
kernel module aacraid Dell PERC2, 2/Si, 3/Si, 3/Di, Adaptec Advanced Raid Products, HP NetRAID-4M, IBM ServeRAID & ICP SCSI driver: 1.2.1
[50877]-custom
kernel module acard_ahci ACard AHCI SATA low-level driver: 1.0
kernel module ad7418 AD7416/17/18 driver: 0.4
kernel module ahci AHCI SATA low-level driver: 3.0
kernel module aic79xx Adaptec AIC790X U320 SCSI Host Bus Adapter driver: 3.0
kernel module aic7xxx Adaptec AIC77XX/78XX SCSI Host Bus Adapter driver: 7.0
kernel module aic94xx Adaptec aic94xx SAS/SATA driver: 1.0.3
kernel module am53c974 AM53C974 SCSI driver: 1.00
kernel module amd_xgbe AMD 10 Gigabit Ethernet Driver: 1.0.3
kernel module arcmsr Areca ARC11xx/12xx/16xx/188x SAS/SATA RAID Controller Driver: v1.40.00.09-20180709
kernel module ata_generic low-level driver for generic ATA: 0.2.15
kernel module ata_piix SCSI low-level driver for Intel PIIX/ICH ATA controllers: 2.13
kernel module atl1 Atheros L1 Gigabit Ethernet Driver: 2.1.3
kernel module atl1c Qualcomm Atheros 100/1000M Ethernet Network Driver: 1.0.1.1-NAPI

kernel module atl1e Atheros 1000M Ethernet Network Driver: 1.0.0.7-NAPI
kernel module atl2 Atheros Fast Ethernet Network Driver: 2.2.3
kernel module atlantic aQuantia Corporation(R) Network Driver: 2.0.3.0-kern
kernel module atxp1 System voltages control via Attansic ATXP1: 0.6.3
kernel module b44 Broadcom 44xx/47xx 10/100 PCI ethernet driver: 2.0
kernel module be2iscsi Emulex OneConnectOpen-iSCSI Driver version11.4.0.1 Driver 11.4.0.1: 11.4.0.1
kernel module be2net Emulex OneConnect NIC Driver 12.0.0.0: 12.0.0.0
kernel module bfa QLogic BR-series Fibre Channel HBA Driver fcpim: 3.2.25.1
kernel module bna QLogic BR-series 10G PCIe Ethernet driver: 3.2.25.1
kernel module bnx2 QLogic BCM5706/5708/5709/5716 Driver: 2.2.6
kernel module bnx2fc QLogic FCoE Driver: 2.11.8
kernel module bnx2i QLogic NetXtreme II BCM5706/5708/5709/57710/57711/57712/57800/57810/57840 iSCSI Driver: 2.7.10.1
kernel module bnx2x QLogic BCM57710/57711/57711E/57712/57712_MF/57800/57800_MF/57810/57810_MF/57840/57840_MF Driver: 1.712.30-0
kernel module bnx_en Broadcom BCM573xx network driver: 1.9.2
kernel module bonding Ethernet Channel Bonding Driver, v3.7.1: 3.7.1
kernel module cnic QLogic cnic Driver: 2.5.22
kernel module csiostor Chelsio FCoE driver: 1.0.0-ko
kernel module cxgb3 Chelsio T3 Network Driver: 1.1.5-ko
kernel module cxgb3i Chelsio T3 iSCSI Driver: 2.0.1-ko
kernel module cxgb4 Chelsio T4/T5/T6 Network Driver: 2.0.0-ko
kernel module cxgb4i Chelsio T4-T6 iSCSI Driver: 0.9.5-ko
kernel module cxgb4vf Chelsio T4/T5/T6 Virtual Function (VF) Network Driver: 2.0.0-ko
kernel module dca : 1.12.1
kernel module dcdbas Dell Systems Management Base Driver (version 5.6.0-3.2): 5.6.0-3.2
kernel module de2104x Intel/Digital 21040/1 series PCI Ethernet driver: 0.7
kernel module dmfe Davicom DM910X fast ethernet driver: 1.36.4
kernel module e100 Intel(R) PRO/100 Network Driver: 3.5.24-k2-NAPI
kernel module e1000 Intel(R) PRO/1000 Network Driver: 7.3.21-k8-NAPI
kernel module e1000e Intel(R) PRO/1000 Network Driver: 3.2.6-k
kernel module eeprom_93cx6 EEPROM 93cx6 chip driver: 1.0
kernel module efivars sysfs interface to EFI Variables: 0.08
kernel module ena Elastic Network Adapter (ENA): 1.5.0K
kernel module enic Cisco VIC Ethernet NIC Driver: 2.3.0.53
kernel module esas2r esas2r: 1.00
kernel module esp_scsi ESP SCSI driver core: 2.000
kernel module fm10k Intel(R) Ethernet Switch Host Interface Driver: 0.23.4-k
kernel module fnic Cisco FCoE HBA Driver: 1.6.0.34
kernel module hpsa Driver for HP Smart Array Controller version 3.4.20-125: 3.4.20-125
kernel module i40e Intel(R) 40-10 Gigabit Ethernet Connection Network Driver: 2.7.29
kernel module i40e Intel(R) Ethernet Connection XL710 Network Driver: 2.3.2-k
kernel module i40evf Intel(R) XL710 X710 Virtual Function Network Driver: 3.2.2-k
kernel module ice Intel(R) Ethernet Connection E800 Series Linux Driver: ice-0.7.0-k
kernel module igb Intel(R) Gigabit Ethernet Network Driver: 5.4.0-k
kernel module igbvf Intel(R) Gigabit Virtual Function Network Driver: 2.4.0-k
kernel module ioatdma : 4.00
kernel module ipmi_msghandler Incoming and outgoing message routing for an IPMI interface.: 39.2
kernel module ipr IBM Power RAID SCSI Adapter Driver: 2.6.4
kernel module ips IBM ServeRAID Adapter Driver 7.12.05: 7.12.05
kernel module iscsi : 1.2.0
kernel module ixgb Intel(R) PRO/10GbE Network Driver: 1.0.135-k2-NAPI
kernel module ixgbe Intel(R) 10 Gigabit PCI Express Network Driver: 5.1.0-k
kernel module ixgbe Intel(R) 10GbE PCI Express Linux Network Driver: 5.5.5
kernel module ixgbevfn Intel(R) 10 Gigabit Virtual Function Network Driver: 4.1.0-k

kernel module jme JMicron JMC2x0 PCI Express Ethernet driver: 1.0.8
 kernel module libata Library module for ATA devices: 3.00
 kernel module libcxgb Chelsio common library: 1.0.0-ko
 kernel module libcxgbi Chelsio iSCSI driver library: 0.9.1-ko
 kernel module liquidio Cavium LiquidIO Intelligent Server Adapter Driver: 1.7.2
 kernel module liquidio_vf Cavium LiquidIO Intelligent Server Adapter Virtual Function Driver: 1.7.2
 kernel module lpfc Emulex LightPulse Fibre Channel SCSI driver 12.0.0.6: 0
 kernel module megaraid LSI Logic MegaRAID legacy driver: 2.00.4
 kernel module megaraid_mbox LSI Logic MegaRAID Mailbox Driver: 2.20.5.1
 kernel module megaraid_mm LSI Logic Management Module: 2.20.2.7
 kernel module megaraid_sas Avago MegaRAID SAS Driver: 07.706.03.00-rc1
 kernel module mlx4_core Mellanox ConnectX HCA low-level driver: 4.0-0
 kernel module mlx4_en Mellanox ConnectX HCA Ethernet driver: 4.0-0
 kernel module mlx5_core Mellanox 5th generation network adapters (ConnectX series) core driver: 5.0-0
 kernel module mpt3sas LSI MPT Fusion SAS 3.0 Device Driver: 26.100.00.00
 kernel module mptbase Fusion MPT base driver: 3.04.20
 kernel module mptctl Fusion MPT misc device (ioctl) driver: 3.04.20
 kernel module mptfc Fusion MPT FC Host driver: 3.04.20
 kernel module mptsas Fusion MPT SAS Host driver: 3.04.20
 kernel module mptscsih Fusion MPT SCSI Host driver: 3.04.20
 kernel module mptspi Fusion MPT SPI Host driver: 3.04.20
 kernel module mtip32xx Micron RealSSD PCIe Block Driver: 1.3.1
 kernel module mvsas Marvell 88SE6440 SAS/SATA controller driver: 0.8.16
 kernel module myri10ge Myricom 10G driver (10GbE): 1.5.3-1.534
 kernel module netxen_nic QLogic/NetXen (1/10) GbE Intelligent Ethernet Driver: 4.0.82
 kernel module nfp The Netronome Flow Processor (NFP) driver.: 4.19.56
 kernel module nicpf Cavium Thunder NIC Physical Function Driver: 1.0
 kernel module nicvf Cavium Thunder NIC Virtual Function Driver: 1.0
 kernel module niu NIU ethernet driver: 1.1
 kernel module nvme : 1.0
 kernel module nvme_core : 1.0
 kernel module pata_acpi SCSI low-level driver for ATA in ACPI mode: 0.2.3
 kernel module pata_ali low-level driver for ALi PATA: 0.7.8
 kernel module pata_amd low-level driver for AMD and Nvidia PATA IDE: 0.4.1
 kernel module pata_artop SCSI low-level driver for ARTOP PATA: 0.4.6
 kernel module pata_atiixp low-level driver for ATI IXP200/300/400: 0.4.6
 kernel module pata_atp867x low level driver for Artop/Acard 867x ATA controller: 0.7.5
 kernel module pata_cmd64x low-level driver for CMD64x series PATA controllers: 0.2.18
 kernel module pata_efar SCSI low-level driver for EFAR PIIX clones: 0.4.5
 kernel module pata_hpt366 low-level driver for the Highpoint HPT366/368: 0.6.11
 kernel module pata_hpt37x low-level driver for the Highpoint HPT37x/30x: 0.6.23
 kernel module pata_hpt3x2n low-level driver for the Highpoint HPT3xxN: 0.3.15
 kernel module pata_hpt3x3 low-level driver for the Highpoint HPT343/363: 0.6.1
 kernel module pata_it821x low-level driver for the IT8211/IT8212 IDE RAID controller: 0.4.2
 kernel module pata_jmicron SCSI low-level driver for Jmicron PATA ports: 0.1.5
 kernel module pata_marvell SCSI low-level driver for Marvell ATA in legacy mode: 0.1.6
 kernel module pata_mpiix low-level driver for Intel MPIIX: 0.7.7
 kernel module pata_netcell SCSI low-level driver for Netcell PATA RAID: 0.1.7
 kernel module pata_ninja32 low-level driver for Ninja32 ATA: 0.1.5
 kernel module pata_ns87410 low-level driver for Nat Semi 87410: 0.4.6
 kernel module pata_ns87415 ATA low-level driver for NS87415 controllers: 0.0.1
 kernel module pata_oldpiix SCSI low-level driver for early PIIX series controllers: 0.5.5
 kernel module pata_pdc2027x libata driver module for Promise PDC20268 to PDC20277: 1.0
 kernel module pata_pdc202xx_old low-level driver for Promise 2024x and 20262-20267: 0.4.3

kernel module pata_platform low-level driver for platform device ATA: 1.2
 kernel module pata_rdc SCSI low-level driver for RDC PATA controllers: 0.01
 kernel module pata_rz1000 low-level driver for RZ1000 PCI ATA: 0.2.4
 kernel module pata_sch SCSI low-level driver for Intel SCH PATA controllers: 0.2
 kernel module pata_serverworks low-level driver for Serverworks OSB4/CSB5/CSB6: 0.4.3
 kernel module pata_sil680 low-level driver for Si680 PATA: 0.4.9
 kernel module pata_sis SCSI low-level driver for SiS ATA: 0.5.2
 kernel module pata_sl82c105 low-level driver for Si82c105: 0.3.3
 kernel module pata_triflex low-level driver for Compaq Triflex: 0.2.8
 kernel module pata_via low-level driver for VIA PATA: 0.3.4
 kernel module pdc_adma Pacific Digital Corporation ADMA low-level driver: 1.0
 kernel module pm80xx PMC-Sierra PM8001/8006/8081/8088/8089/8074/8076/8077/8070/8072 SAS/SATA controller driver: 0.1.38
 kernel module pmcraid PMC Sierra MaxRAID Controller Driver: 1.0.3
 kernel module qed QLogic FastLinQ 4xxxx Core Module: 8.33.0.20
 kernel module qede QLogic FastLinQ 4xxxx Ethernet Driver: 8.33.0.20
 kernel module qedf QLogic QEDF 25/40/50/100Gb FCoE Driver: 8.33.16.20
 kernel module qedi QLogic FastLinQ 4xxxx iSCSI Module: 8.33.0.20
 kernel module qla1280 Qlogic ISP SCSI (qla1x80/qla1x160) driver: 3.27.1
 kernel module qla2xxx QLogic Fibre Channel HBA Driver: 10.00.00.08-k
 kernel module qla3xxx QLogic ISP3XXX Network Driver v2.03.00-k5 : v2.03.00-k5
 kernel module qla4xxx QLogic iSCSI HBA Driver: 5.04.00-k6
 kernel module qlcnic QLogic 1/10 GbE Converged/Intelligent Ethernet Driver: 5.3.66
 kernel module qlge QLogic 10 Gigabit PCI-E Ethernet Driver : 1.00.00.35
 kernel module r6040 RDC R6040 NAPI PCI FastEthernet driver: 0.29 04Jul2016
 kernel module rsxx IBM Flash Adapter 900GB Full Height Device Driver: 4.0.3.2516
 kernel module s2io : 2.0.26.28
 kernel module sata_dwc_460ex DesignWare Cores SATA controller low level driver: 1.3
 kernel module sata_mv SCSI low-level driver for Marvell SATA controllers: 1.28
 kernel module sata_nv low-level driver for NVIDIA nForce SATA controller: 3.5
 kernel module sata_promise Promise ATA TX2/TX4/TX4000 low-level driver: 2.12
 kernel module sata_qstor Pacific Digital Corporation QStor SATA low-level driver: 0.09
 kernel module sata_sil low-level driver for Silicon Image SATA controller: 2.4
 kernel module sata_sis low-level driver for Silicon Integrated Systems SATA controller: 1.0
 kernel module sata_svw low-level driver for K2 SATA controller: 2.3
 kernel module sata_sx4 Promise SATA low-level driver: 0.12
 kernel module sata_uli low-level driver for ULI Electronics SATA controller: 1.3
 kernel module sata_via SCSI low-level driver for VIA SATA controllers: 2.6
 kernel module sata_vsc low-level driver for Vitesse VSC7174 SATA controller: 2.3
 kernel module sfc Solarflare network driver: 4.1
 kernel module sfc_falcon Solarflare Falcon network driver: 4.1
 kernel module sg SCSI generic (sg) driver: 3.5.36
 kernel module sis190 SiS sis190/191 Gigabit Ethernet driver: 1.4
 kernel module skge SysKonnect Gigabit Ethernet driver: 1.14
 kernel module sky2 Marvell Yukon 2 Gigabit Ethernet driver: 1.30
 kernel module slicoss Alacritech non-accelerated SLIC driver: 1.0
 kernel module smartpqi Driver for Microsemi Smart Family Controller version 1.1.4-130: 1.1.4-130
 kernel module smsc911x : 2008-10-21
 kernel module smsc9420 : 1.01
 kernel module snic Cisco SCSI NIC Driver: 0.0.1.18
 kernel module starfire Adaptec Starfire Ethernet driver: 2.1
 kernel module stex Promise Technology SuperTrak EX Controllers: 6.02.0000.01
 kernel module sunhme Sun HappyMealEthernet(HME) 10/100baseT ethernet driver: 3.10
 kernel module sym53c8xx NCR, Symbios and LSI 8xx and 1010 PCI SCSI adapters: 2.2.3
 kernel module tg3 Broadcom Tigon3 ethernet driver: 3.137

kernel module thunder_bgx Cavium Thunder BGX/MAC Driver: 1.0
kernel module thunder_xcv Cavium Thunder RGX/XCV Driver: 1.0
kernel module tpm TPM Driver: 2.0
kernel module tpm_atmel TPM Driver: 2.0
kernel module tpm_crb TPM2 Driver: 0.1
kernel module tpm_i2c_infineon TPM TIS I2C Infineon Driver: 2.2.0
kernel module tpm_infineon Driver for Infineon TPM SLD 9630 TT 1.1 / SLB 9635 TT 1.2: 1.9.2
kernel module tpm_nsc TPM Driver: 2.0
kernel module tpm_st33zp24 ST33ZP24 TPM 1.2 driver: 1.3.0
kernel module tpm_st33zp24_i2c STM TPM 1.2 I2C ST33 Driver: 1.3.0
kernel module tpm_tis TPM Driver: 2.0
kernel module tpm_tis_core TPM Driver: 2.0
kernel module tpm_vtpm_proxy vTPM Driver: 0.1
kernel module tulip Digital 21*4* Tulip ethernet driver: 1.1.15
kernel module typhoon 3Com Typhoon Family (3C990, 3CR990, and variants): 1.0
kernel module ufshcd_core Generic UFS host controller driver Core: 0.2
kernel module virtio_pci virtio-pci: 1
kernel module vmw_pvscsi VMware PVSCSI driver: 1.0.7.0-k
kernel module vmxnet3 VMware vmxnet3 virtual NIC driver: 1.4.16.0-k
kernel module vxlan Driver for VXLAN encapsulated traffic: 0.1
kernel module winbond_840 Winbond W89c840 Ethernet driver: 1.01-e
adduser version: 3.115
apt version: 1.4.9
apt-utils version: 1.4.9
base-files version: 9.9+deb9u9
base-passwd version: 3.5.43
bash version: 4.4-5
bsdmainutils version: 9.0.12+nmu1
bsdutils version: 1:2.29.2-1+deb9u1
bzip2 version: 1.0.6-8.1
ca-certificates version: 20161130+nmu1+deb9u1
coreutils version: 8.26-3
cpio version: 2.11+dfsg-6
cron version: 3.0pl1-128+deb9u1
cryptsetup-bin version: 2:1.7.3-4
curl version: 7.52.1-5+deb9u9
dash version: 0.5.8-2.4
dbus version: 1.10.26-0+deb9u1
debconf version: 1.5.61
debconf-i18n version: 1.5.61
debian-archive-keyring version: 2017.5
debiantutils version: 4.8.1.1
diffutils version: 1:3.5-3
dmidecode version: 3.0-4
dmsetup version: 2:1.02.137-2
dosfstools version: 4.1-1
dpkg version: 1.18.25
e2fslibs version: 1.43.4-2
e2fsprogs version: 1.43.4-2
ethtool version: 1:4.8-1+b1
file version: 1:5.30-1+deb9u2
findutils version: 4.6.0+git+20161106-2
gcc-6-base version: 6.3.0-18+deb9u1
gdisk version: 1.0.1-1

gnupg version: 2.1.18-8~deb9u4
gnupg-agent version: 2.1.18-8~deb9u4
gpgv version: 2.1.18-8~deb9u4
grep version: 2.27-2
groff-base version: 1.22.3-9
guile-2.0-libs version: 2.0.13+1-4
gzip version: 1.6-5+b1
hdparm version: 9.51+ds-1+deb9u1
hostname version: 3.18+b1
hwinfo version: 21.38-1
ifenslave version: 2.9
ifenslave-2.6 version: 2.9
ifupdown version: 0.8.19
init version: 1.48
init-system-helpers version: 1.48
iperf3 version: 3.1.3-1
iproute2 version: 4.9.0-1+deb9u1
iptables version: 1.6.0+snapshot20161117-6
iputils-ping version: 3:20161105-1
irqbalance version: 1.1.0-2.3
isc-dhcp-client version: 4.3.5-3+deb9u1
isc-dhcp-common version: 4.3.5-3+deb9u1
kmod version: 23-2
krb5-locales version: 1.15-1+deb9u1
less version: 481-2.1
libacl1 version: 2.2.52-3+b1
libapparmor1 version: 2.11.0-3+deb9u2
libapt-inst2.0 version: 1.4.9
libapt-pkg5.0 version: 1.4.9
libassuan0 version: 2.4.3-2
libattr1 version: 1:2.4.47-2+b2
libaudit-common version: 1:2.6.7-2
libaudit1 version: 1:2.6.7-2
libblkid1 version: 2.29.2-1+deb9u1
libboost-python1.62.0 version: 1.62.0+dfsg-4
libboost-system1.62.0 version: 1.62.0+dfsg-4
libboost-thread1.62.0 version: 1.62.0+dfsg-4
libbsd0 version: 0.8.3-1
libbz2-1.0 version: 1.0.6-8.1
libc-bin version: 2.24-11+deb9u4
libc6 version: 2.24-11+deb9u4
libcap-ng0 version: 0.7.7-3+b1
libcap2 version: 1:2.25-1
libcomerr2 version: 1.43.4-2
libcryptsetup4 version: 2:1.7.3-4
libcurl3 version: 7.52.1-5+deb9u9
libdb5.3 version: 5.3.28-12+deb9u1
libdbus-1-3 version: 1.10.26-0+deb9u1
libdebconfclient0 version: 0.227
libdevmapper1.02.1 version: 2:1.02.137-2
libdns-export162 version: 1:9.10.3.dfsg.P4-12.3+deb9u4
libedit2 version: 3.1-20160903-3
libelf1 version: 0.168-1
libestr0 version: 0.1.10-2

libexpat1 version: 2.2.0-2+deb9u1
libfastjson4 version: 0.99.4-1
libfdisk1 version: 2.29.2-1+deb9u1
libffi6 version: 3.2.1-6
libfribidi0 version: 0.19.7-1+b1
libgc1c2 version: 1:7.4.2-8
libgcc1 version: 1:6.3.0-18+deb9u1
libgcrypt20 version: 1.7.6-2+deb9u3
libgdbm3 version: 1.8.3-14
libglib2.0-0 version: 2.50.3-2
libglib2.0-data version: 2.50.3-2
libgmp10 version: 2:6.1.2+dfsg-1
libgnutls-openssl27 version: 3.5.8-5+deb9u4
libgnutls30 version: 3.5.8-5+deb9u4
libgpg-error0 version: 1.26-2
libgsasl7 version: 1.8.0-8+b2
libgssapi-krb5-2 version: 1.15-1+deb9u1
libhd21 version: 21.38-1
libhogweed4 version: 3.3-1+b2
libicu57 version: 57.1-6+deb9u2
libidn11 version: 1.33-1
libidn2-0 version: 0.16-1+deb9u1
libip4tc0 version: 1.6.0+snapshot20161117-6
libip6tc0 version: 1.6.0+snapshot20161117-6
libiperf0 version: 3.1.3-1
libiptc0 version: 1.6.0+snapshot20161117-6
libisc-export160 version: 1:9.10.3.dfsg.P4-12.3+deb9u4
libk5crypto3 version: 1.15-1+deb9u1
libkeyutils1 version: 1.5.9-9
libkmod2 version: 23-2
libkrb5-3 version: 1.15-1+deb9u1
libkrb5support0 version: 1.15-1+deb9u1
libksba8 version: 1.3.5-2
libkyotocabinet16v5 version: 1.2.76-4.2+b1
libldap-2.4-2 version: 2.4.44+dfsg-5+deb9u2
libldap-common version: 2.4.44+dfsg-5+deb9u2
liblocale-gettext-perl version: 1.07-3+b1
liblogging-stdlog0 version: 1.0.5-2+b2
liblognorm5 version: 2.0.1-1.1+b1
libltdl7 version: 2.4.6-2
liblz4-1 version: 0.0~r131-2+b1
liblzma5 version: 5.2.2-1.2+b1
liblzo2-2 version: 2.08-1.2+b2
libmagic-mgc version: 1:5.30-1+deb9u2
libmagic1 version: 1:5.30-1+deb9u2
libmailutils5 version: 1:3.1.1-1
libmariadbclient18 version: 10.1.38-0+deb9u1
libmnl0 version: 1.0.4-2
libmount1 version: 2.29.2-1+deb9u1
libncurses5 version: 6.0+20161126-1+deb9u2
libncursesw5 version: 6.0+20161126-1+deb9u2
libnetfilter-contrack3 version: 1.0.6-2
libnettle6 version: 3.3-1+b2
libnewt0.52 version: 0.52.19-1+b1

libnfnetlink0 version: 1.0.1-3
libnghttp2-14 version: 1.18.1-1
libnpt0 version: 1.3-1
libntlm0 version: 1.4-8
libnuma1 version: 2.0.11-2.1
libopenipmi0 version: 2.0.22-1.1
libopts25 version: 1:5.18.12-3
libp11-kit0 version: 0.23.3-2
libpam-modules version: 1.1.8-3.6
libpam-modules-bin version: 1.1.8-3.6
libpam-runtime version: 1.1.8-3.6
libpam-systemd version: 232-25+deb9u11
libpam0g version: 1.1.8-3.6
libpci3 version: 1:3.5.2-1
libpcre3 version: 2:8.39-3
libperl5.24 version: 5.24.1-3+deb9u5
libpipeline1 version: 1.4.1-2
libpopt0 version: 1.16-10+b2
libprocps6 version: 2:3.3.12-3+deb9u1
libpsl5 version: 0.17.0-3
libpython2.7 version: 2.7.13-2+deb9u3
libpython2.7-minimal version: 2.7.13-2+deb9u3
libpython2.7-stdlib version: 2.7.13-2+deb9u3
libreadline7 version: 7.0-3
librtmp1 version: 2.4+20151223.gitfa8646d.1-1+b1
libsasl2-2 version: 2.1.27~101-g0780600+dfsg-3
libsasl2-modules version: 2.1.27~101-g0780600+dfsg-3
libsasl2-modules-db version: 2.1.27~101-g0780600+dfsg-3
libseccomp2 version: 2.3.1-2.1+deb9u1
libselinux1 version: 2.6-3+b3
libsemanage-common version: 2.6-2
libsemanage1 version: 2.6-2
libsensors4 version: 1:3.4.0-4
libsepol1 version: 2.6-2
libsgutils2-2 version: 1.42-2
libslang2 version: 2.3.1-5
libsmartcols1 version: 2.29.2-1+deb9u1
libsnmp-base version: 5.7.3+dfsg-1.7+deb9u1
libsnmp30 version: 5.7.3+dfsg-1.7+deb9u1
libsqlite3-0 version: 3.16.2-5+deb9u1
libss2 version: 1.43.4-2
libssh2-1 version: 1.7.0-1+deb9u1
libssl1.0.2 version: 1.0.2r-1~deb9u1
libssl1.1 version: 1.1.0j-1~deb9u1
libstdc++6 version: 6.3.0-18+deb9u1
libsysfs2 version: 2.1.0+repack-4+b2
libsystemd0 version: 232-25+deb9u11
libtasn1-6 version: 4.10-1.1+deb9u1
libtext-charwidth-perl version: 0.04-7+b5
libtext-iconv-perl version: 1.7-5+b4
libtext-wrapi18n-perl version: 0.06-7.1
libtinfo5 version: 6.0+20161126-1+deb9u2
libudev1 version: 232-25+deb9u11
libunistring0 version: 0.9.6+really0.9.3-0.1

libustr-1.0-1 version: 1.0.4-6
libuuid1 version: 2.29.2-1+deb9u1
libwrap0 version: 7.6.q-26
libx11-6 version: 2:1.6.4-3+deb9u1
libx11-data version: 2:1.6.4-3+deb9u1
libx86emu1 version: 1.11-2
libxapian30 version: 1.4.3-2+deb9u3
libxau6 version: 1:1.0.8-1
libxcb1 version: 1.12-1
libxdmcp6 version: 1:1.1.2-3
libxext6 version: 2:1.3.3-1+b2
libxml2 version: 2.9.4+dfsg1-2.2+deb9u2
libxmu1 version: 2:1.1.2-2
libxtables12 version: 1.6.0+snapshot20161117-6
linux-firmware version: 1.179
login version: 1:4.4-4.1
logrotate version: 3.11.0-0.1
lsb-base version: 9.20161125
lsscsi version: 0.27-3+b1
mailutils version: 1:3.1.1-1
mailutils-common version: 1:3.1.1-1
mawk version: 1.3.3-17+b3
megacli version: 8.07.14-2
mime-support version: 3.60
mount version: 2.29.2-1+deb9u1
multiarch-support version: 2.24-11+deb9u4
mysql-common version: 5.8+1.0.2
nano version: 2.7.4-1
ncurses-base version: 6.0+20161126-1+deb9u2
ncurses-bin version: 6.0+20161126-1+deb9u2
ncurses-term version: 6.0+20161126-1+deb9u2
net-tools version: 1.60+git20161116.90da8a0-1
netbase version: 5.4
nload version: 0.7.4-1+b2
ntp version: 1:4.2.8p10+dfsg-3+deb9u2
numactl version: 2.0.11-2.1
openipmi version: 2.0.22-1.1
openssh-client version: 1:7.4p1-10+deb9u6
openssh-server version: 1:7.4p1-10+deb9u6
openssh-sftp-server version: 1:7.4p1-10+deb9u6
openssl version: 1.1.0j-1~deb9u1
passwd version: 1:4.4-4.1
perl version: 5.24.1-3+deb9u5
perl-base version: 5.24.1-3+deb9u5
perl-modules-5.24 version: 5.24.1-3+deb9u5
pinentry-curses version: 1.0.0-2
powermgmt-base version: 1.31+nmu1
procps version: 2:3.3.12-3+deb9u1
python2.7 version: 2.7.13-2+deb9u3
python2.7-minimal version: 2.7.13-2+deb9u3
readline-common version: 7.0-3
rename version: 0.20-4
rsyslog version: 8.24.0-1
sdparm version: 1.08-1+b1

sed version: 4.4-1
sensible-utils version: 0.0.9+deb9u1
sg3-utils version: 1.42-2
sgml-base version: 1.29
shared-mime-info version: 1.8-1+deb9u1
smartmontools version: 6.5+svn4324-1
smp-utils version: 0.98-1
snmp version: 5.7.3+dfsg-1.7+deb9u1
snmpd version: 5.7.3+dfsg-1.7+deb9u1
ssh version: 1:7.4p1-10+deb9u6
ssmtp version: 2.64-8+b2
sysstat version: 11.4.3-2
systemd version: 232-25+deb9u11
systemd-sysv version: 232-25+deb9u11
sysvinit-utils version: 2.88dsf-59.9
tar version: 1.29b-1.1
tasksel version: 3.39
tasksel-data version: 3.39
tcpd version: 7.6.q-26
toftpdos version: 1.7.13+ds-2
traceroute version: 1:2.1.0-2
tzdata version: 2019a-0+deb9u1
ucf version: 3.0036
udev version: 232-25+deb9u11
util-linux version: 2.29.2-1+deb9u1
vim-common version: 2:8.0.0197-4+deb9u1
vim-tiny version: 2:8.0.0197-4+deb9u1
wget version: 1.18-5+deb9u3
whiptail version: 0.52.19-1+b1
xauth version: 1:1.0.9-1+b2
xdg-user-dirs version: 0.15-2+b1
xml-core version: 0.17
xxd version: 2:8.0.0197-4+deb9u1
xz-utils version: 5.2.2-1.2+b1
zlib1g version: 1:1.2.8.dfsg-5
iperf3: 3.1.3-1
libiperf0: 3.1.3-1
linux-firmware: 1.179
megacli: 8.07.14-2

Additional Elasticsearch-specific Caringo distributions:

Elasticsearch: 5.6.12
elasticsearch-curator: 4.1.2
elasticsearch-py: 2.2.0
click: 6.2
txes: 0.1.4+
urllib3: 1.12

Storage UI Release Notes

If you are upgrading from a prior version, review the release notes for each version since the version from which you are upgrading.

For upgrade steps, see [Installing the Storage UI](#).

- [Storage UI 2 Release](#)
- [Storage UI 3 Release](#)

Storage UI 2 Release

- [Changes in Storage UI 2.3](#)
- [Changes in Storage UI 2.2](#)
- [Changes in Storage UI 2.1](#)
- [Changes in Storage UI 2.0](#)

Changes in Storage UI 2.3

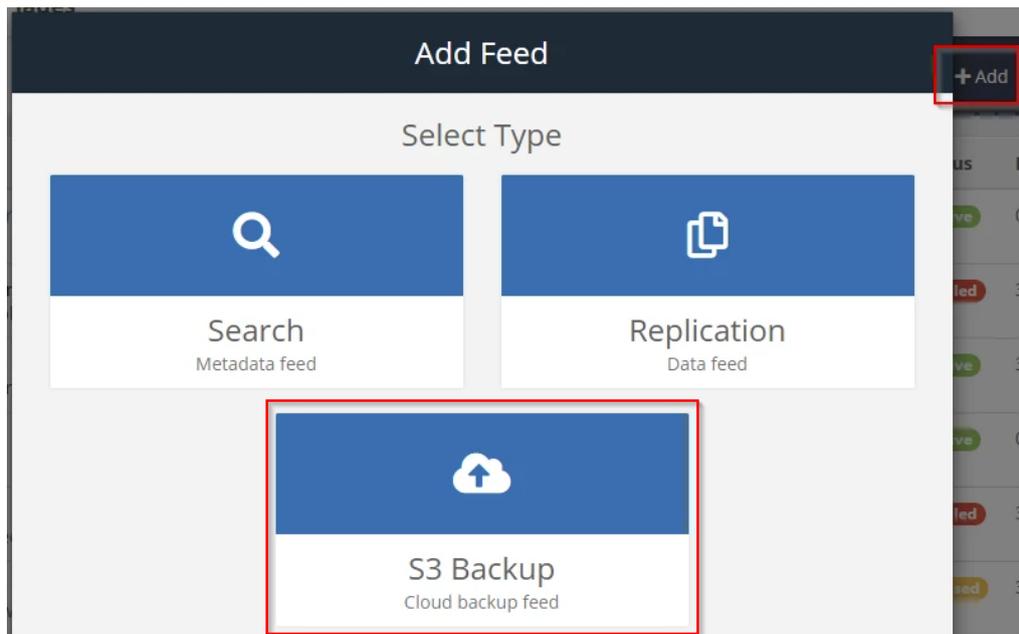
Required integrations

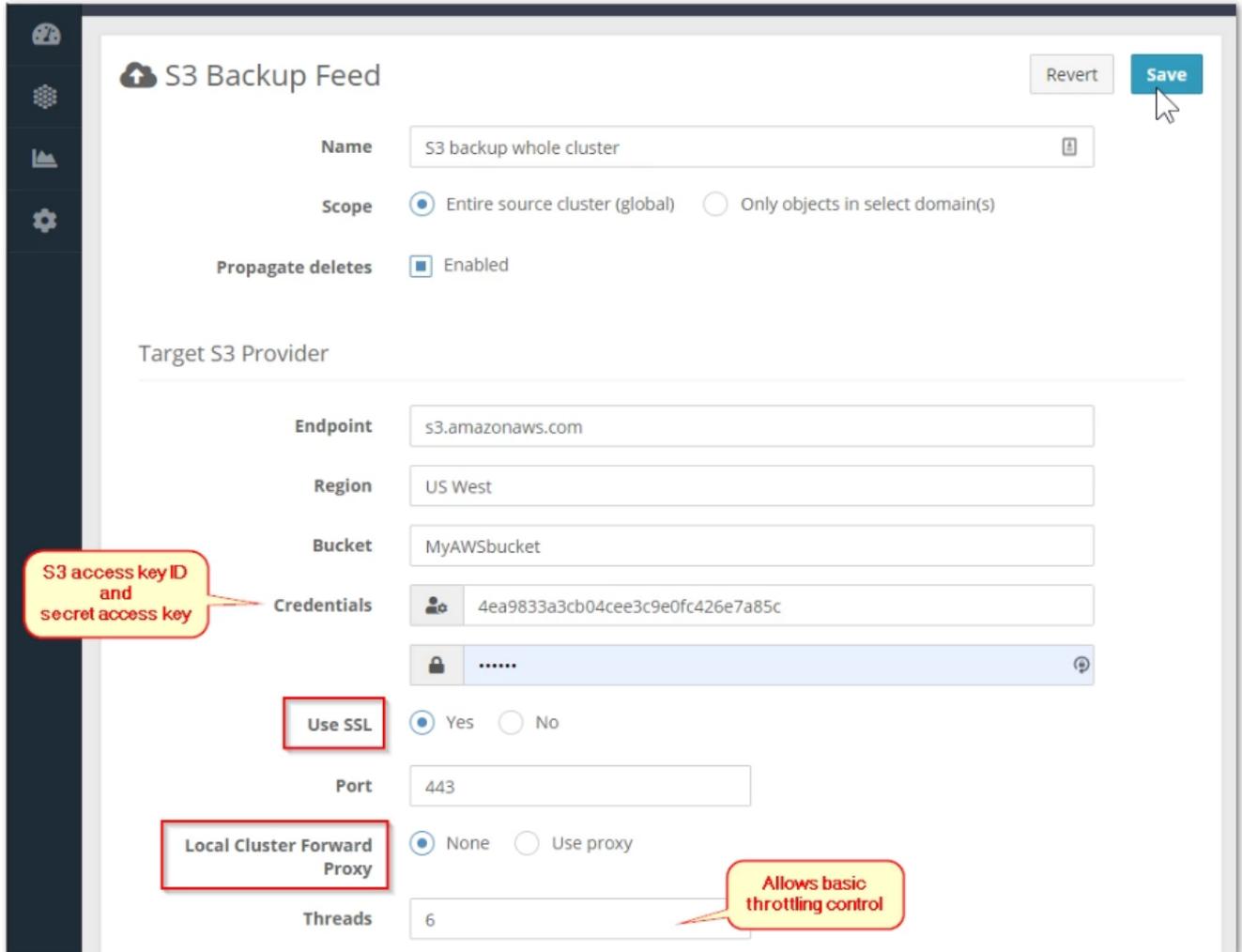
This version requires Gateway version 6.2 or later to use S3 Feeds, and SwarmFS version 2.3 or later, if used.

This release features support for Swarm's new S3 Backup feed type and several new settings for managing SwarmFS behavior and performance in different implementations.

S3 Backup Feeds – Swarm allows tiering to public cloud services for easy off-premises storage for disaster recovery (DR). Amazon S3 has the widest support in the industry, so S3-compatible endpoints are the first cloud destination from Swarm. (UIS-1027)

- On the **Feeds** page of the Swarm UI, S3 Backup feed can be added, which targets an existing S3 bucket. See [S3 Backup Feeds](#).





S3 Backup Feed Revert Save

Name

Scope Entire source cluster (global) Only objects in select domain(s)

Propagate deletes Enabled

Target S3 Provider

Endpoint

Region

Bucket

Credentials

Use SSL Yes No

Port

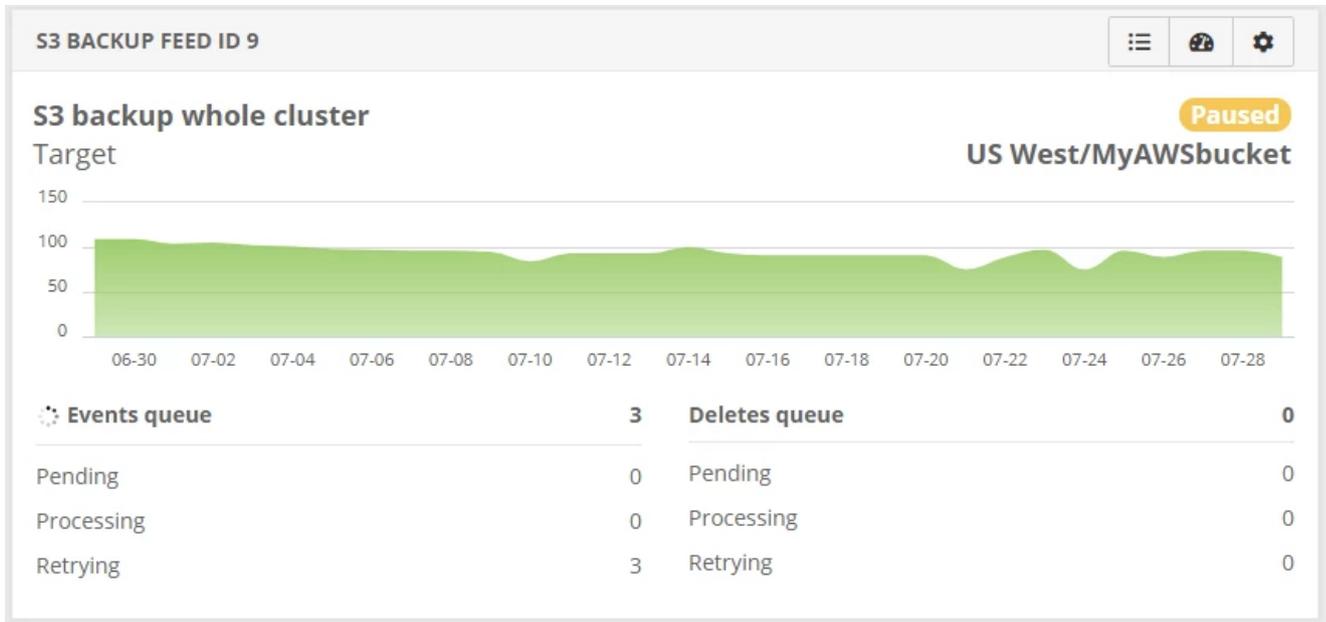
Local Cluster Forward Proxy None Use proxy

Threads

S3 access key ID and secret access key

Allows basic throttling control

- The S3 feed operates like a Replication feed, and it populates the same real-time dashboard charts and feed reports for monitoring backups. (UIS-1027)



SwarmFS Tuning – For SwarmFS exports, several new **Advanced** settings are available to adjust SwarmFS for the environment:

- **Read buffer size** (ReadaheadSize) allows matching the expected workload on a specific share, lowering for small and non-sequential reads, increasing for large and sequential ones. (UIS-1007)
- **Parallel read buffer requests** (ReadaheadCount) allows tuning the performance of large object reads; the default of 4 reflects the optimal number of threads, per performance testing. (UIS-1007)
- **Maximum part size** (MaxPartSize) allows increasing the part size for large (multipart) uploads to improve the throughput when applications are writing huge files. (UIS-1018)
- **Collector sleep time** (CollectorSleepTime) allows minimizing object consolidation by sending fewer and larger sets of data to Swarm (at the expense of both RAM and read performance) if an implementation is sensitive to how quickly the Swarm health processor consolidates objects, which cannot be guaranteed. (UIS-1018)
- **Elasticsearch buffer refresh time** (ESBufferRefreshTime) allows tuning how rapidly non-SwarmFS object updates are reflected in SwarmFS listings. Lower to reduce the wait for consistency, at the cost of increased load on Elasticsearch. (UIS-1037)

In addition, these issues are fixed:

- Issues existed with feeds defined to use a non-default admin password. (UIS-759)
- Clicking **Add Export** on the NFS page caused an immediate 500 error if a cluster had no search feed defined. (UIS-441)

Watch Items and Known Issues

- Feeds can be created but they cannot be updated via Swarm UI 2.3 because of changes to handling of feed definitions when using Gateway 5.4. (UIS-1033)
- On the Chassis Details page, Advanced tab, the data for Health Data and View the raw JSON may not display. (UIS-1048)
- Turn off the identify function before removing the disk from the chassis. Failure to perform this can result in the need to restart the chassis when using the Swarm UI to identify volumes. (UIS-564)
- The UI does not convey pausing cannot begin until the feed backlog is cleared, which can be a long delay when pausing a feed. (UIS-437)

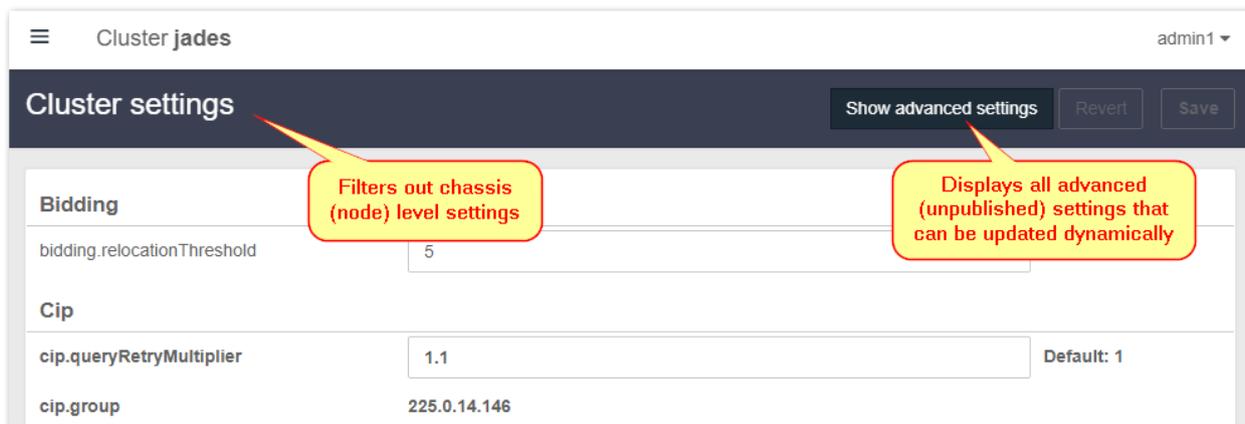
Changes in Storage UI 2.2

Required integrations

This version requires Gateway 5.4 or higher at minimum and Gateway 6.2 to use S3 Feeds, and SwarmFS version 2.2 or later, if used.

This release features improvements to the handling of Swarm Storage settings and how they are accessed in the UI.

- Swarm settings include both cluster-wide and node-specific options, which can vary from chassis to chassis. On the **Cluster Settings** page, the Swarm UI now prevents erroneous changes by hiding the node-specific settings when Platform Server is not implemented and handling those nodes. (UIS-1000)
- The Cluster Settings page now includes an option to **Show advanced settings**. This option reveals all advanced (unpublished) settings that are dynamic (persisted) when enabled. They can be updated on a running cluster without a reboot. As before, bold fonts and Default information displays alert to settings with custom values in the cluster. (UIS-998)



Watch Items and Known Issues

- Issues exist with feeds defined to use a non-default admin password. (UIS-759)
- Turn off the identify function before removing the disk from the chassis. Failure to perform this can result in the need to restart the chassis when using the Swarm UI to identify volumes. (UIS-564)
- Clicking **Add Export** on the NFS page causes an immediate 500 error if a cluster has no search feed defined. (UIS-441)
- Turn off the identify function before removing the disk from the chassis. Failure to perform this can result in the need to restart the chassis when pausing a feed, which can be a long delay. (UIS-437)

Changes in Storage UI 2.1

Required integrations

This version requires Gateway version 5.4 or later and SwarmFS version 2.1 or later, if used.

- Improved: Swarm settings include both cluster-wide and node-specific options. On the Cluster Settings page, the Swarm UI now prevents erroneous changes by hiding the node-specific settings when Platform Server is not implemented and handling those nodes. (UIS-798)

- Improved: Turn off the identify function before removing the disk from the chassis. Failure to perform this so can result in the need to restart the chassis when access tokens expire or are deleted during an active session. (UIS-975)
- Fixed: The UI refreshes and reselects them all if deselecting nodes from the queue without Restarting when editing the Rolling Restart Queue. (UIS-957)
- Fixed: On the Details tab of the Chassis Details page, the Actions menu for each disk erroneously but harmlessly showed a testing-only command to Fail the disk. (UIS-955)

Watch Items and Known Issues

- A node may become temporarily unresponsive when attempting to view **Health Data** (the raw JSON of the health report) on the Advanced tab of the Chassis Details page. (SWAR-8349)
- A storage node may be reported to be in an unknown state rather than in maintenance mode while a reboot is in progress. (SWAR-8348)
- Issues exist with feeds defined to use a non-default admin password. (UIS-759)
- Turn off the identify function before removing the disk from the chassis when using the Swarm UI to identify volumes. Failure to perform this can result in the need to restart the chassis. (UIS-564)
- Clicking **Add Export** on the NFS page causes an immediate 500 error if a cluster has no search feed defined. (UIS-441)
- The UI does not convey pausing cannot begin until the feed backlog is cleared when pausing a feed, which can be a long delay. (UIS-437)

Changes in Storage UI 2.0



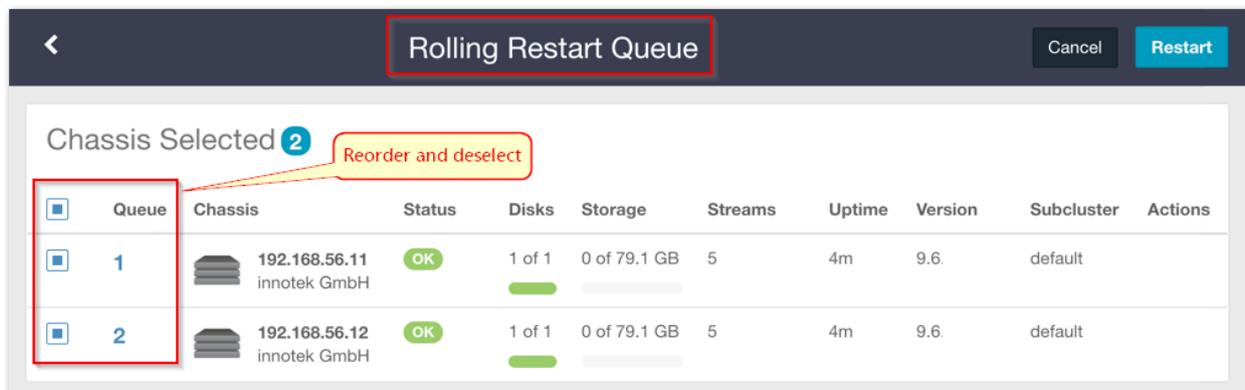
Required integrations

This version requires Gateway version 5.4 or later and SwarmFS version 2.1 or later, if used.

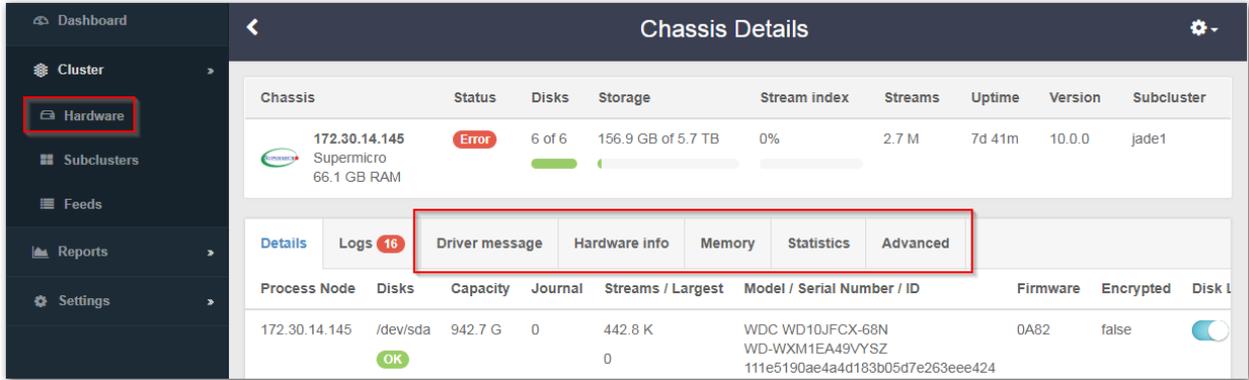
Hardware Management

Aligning with the new architecture of Swarm Storage 10, Storage UI 2.0 is extensively expanded to support the monitoring and administration of Swarm implementations, replicating the rich functionality of the legacy Admin Console (which is deprecated) and adding visibility in to Swarm's management API.

Rolling Restarts – With Platform Server installed, choose to perform a Rolling Restart, so the cluster remains fully operational, with chassis going offline one at a time to avoid service interruption. The Rolling Restart Queue allows reordering and removing chassis from the queue, monitor the progress, and cancel queued restarts. (UIS-588)



Chassis Details page features:



• **Details Tab**

- The main **Details** tab includes counts of each disk's streams and the size of the largest stream. Watching these counts helps monitor the progress of disks being retired. (UIS-533)
- Disks can be retired individually as well as retiring the machine (chassis) as a whole. Disk-level retires are useful for targeting bad (slow) disks and for working around having too limited capacity for retires of entire chassis. (UIS-544)
- The affected chassis now shows the status when one or more disks is in the process of retiring, for improved visibility and tracking. (UIS-749)

• **Logs Tab**

- The **Logs** tab allows clearing out the logs when no longer needed. (UIS-616)

• **Driver Message Tab** (*new*)

- Driver Message displays the output from the **dmesg** command, which prints the message buffer of the kernel. (UIS-799)

• **Hardware Info Tab** (*new*)

- Hardware Info displays the output of the **hwinfo** hardware detection tool. (UIS-799)

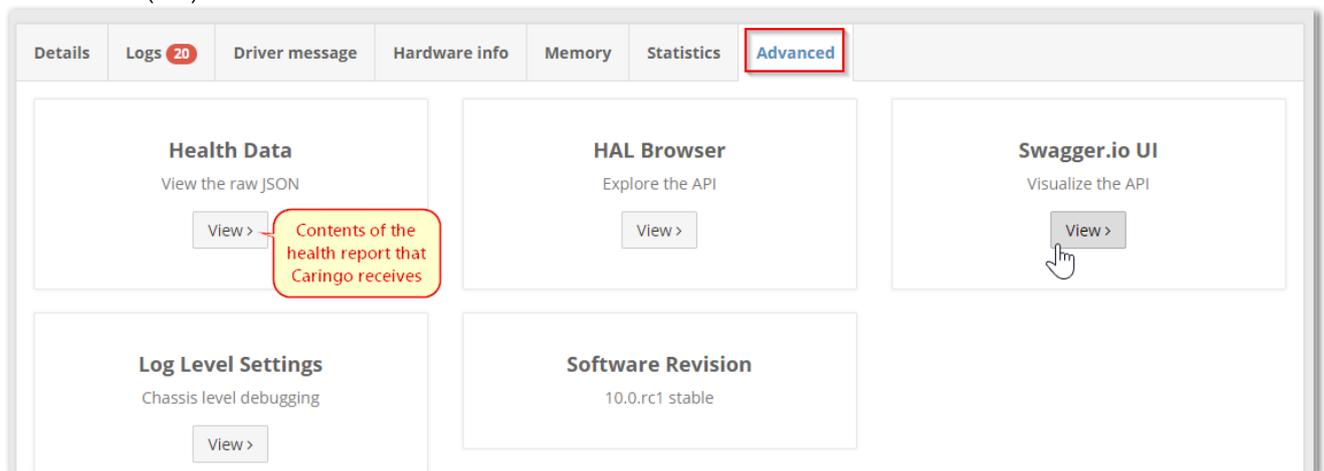
• **Memory Tab** (*new*)

- The Memory tab reports details of memory usage on the specific machine, to help with capacity planning and analysis. (UIS-723)

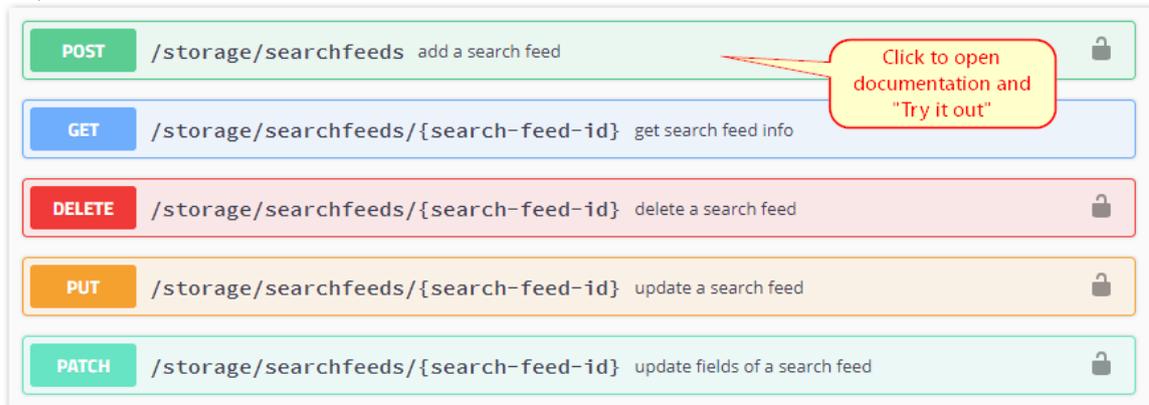
• **Statistics Tab** (*new*)

- The **Statistics** tab rolls up a detailed, expandable report combining Health Processor (HP), Communications (cluster network), and Memory usage counts and values, to help with analysis and troubleshooting. (UIS-886)

• **Advanced Tab** (*new*)



- View the cluster's **Health Data**, which is the raw JSON content of the health report a cluster sends to DataCore Support. (UIS-838)
- View and change the **Log Level Settings** dynamically, to simplify machine-level debugging. (UIS-835)
- Verify the **Software Revision** of Swarm Storage running on the specific machine, which is useful for managing rolling upgrades. (UIS-802)
- Use an embedded **HAL Browser** to explore the complete API for Swarm storage management dynamically. (UIS-836)
- Access the **Swagger.io** UI. This API visualization tool also allows exploring Swarm's API for managing the storage cluster. (UIS-837)



• **Dashb**

- The dashboard now dynamically reports the amount of space available as well as space used in the storage cluster. (UIS-765)

Replication Feeds with SSL

With 2.0, **Replicate via direct POST** now supports SSL/TLS network encryption and standard proxy servers for replication feeds, which eliminates the need for separate VPN tunnels between clusters. This capability streamlines deployments where encrypted communications are needed over wide-area, untrusted networks. See [Replicating Feeds over Untrusted Networks](#).

- Swarm now supports using SSL for remote replication data transfer. This configuration requires an SSL offload proxy (such as HAProxy) in the target cluster environment. (SWAR-7826)

Replication Mode

Replicate via bidirectional GET (if needed)

Replicate via direct POST (recommended; supports Gateway)

Threads

SSL Server

None Require trusted SSL Allow untrusted SSL (not recommended)

Local Cluster Forward Proxy

None Use proxy

Host

Port

Forward Proxy Username and Password

Optional use of proxy on target cluster

Optional addition of forward proxy on source cluster

- For sites using SSL with remote replication, Swarm allows the establishment of self-signed trusted certificates (public keys). (SWAR-8080) See [Adding a Trusted Certificate to Swarm](#).
- Swarm allows placing a forward proxy to the source cluster into the replication path. (SWAR-8025)

Feed Control and Monitoring

- The statuses reported on the **Cluster > Feeds** and **Reports > Feeds** pages now refresh automatically. (UIS-796)
- **Feed table** – To support feed troubleshooting, the Feed Settings page for a given feed now includes a command to **View feed table**, which displays the the SNMP repository dump for the selected node. (UIS-787)
- **Domain filtering** – Filter which domains to include, exclude, or both when defining new replication feeds, and specify whether to replicate any unnamed objects not tenanted in any domain. The domain filters support wildcard matching for ease of maintenance. (UIS-709)

Name

Scope Only objects in select domain(s)

Include domains

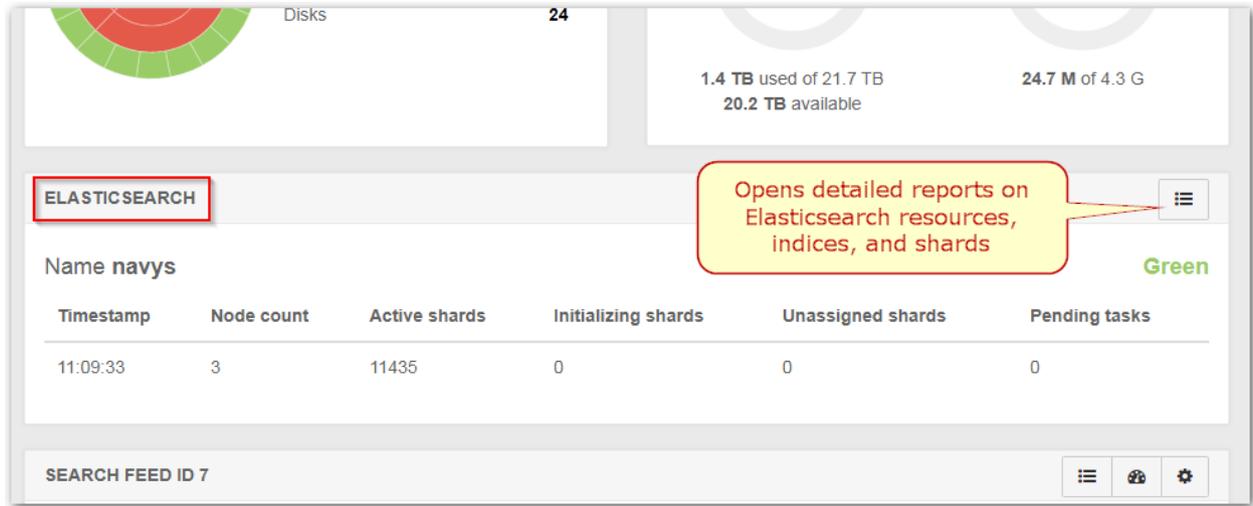
Exclude domains

Include objects without a domain

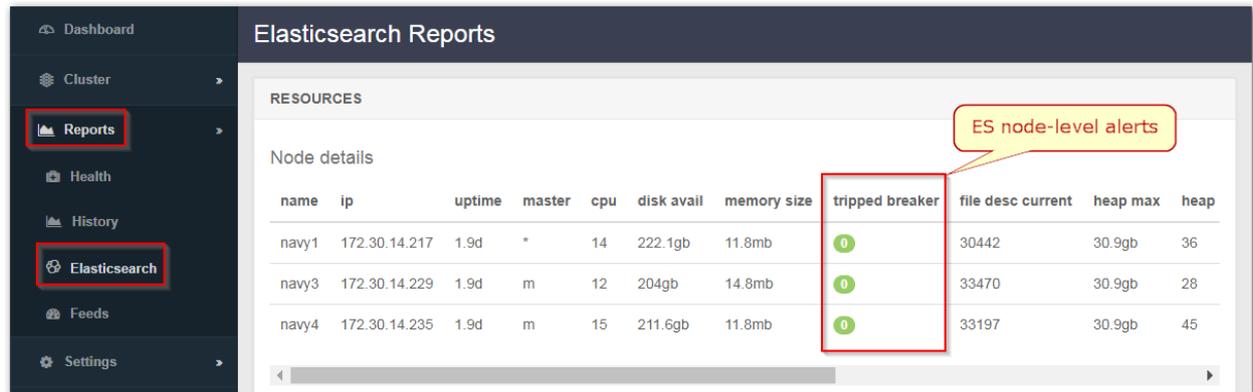
Domain filtering for replication feeds

Elasticsearch

Research the ES cluster status on the Elasticsearch Reports page if the **Elasticsearch** panel on the Dashboard shows a problem.



These reports generate on demand and allow drilling in to details spanning the ES nodes, thread pools, indices, and shards. See [Using Cluster Reports](#).



Additional Changes

- Improved: The cluster-level **Health Report** now totals stream counts for individual machines (chassis) and for the entire cluster. (UIS-716)
- Improved: The Swarm UI now supports being served on ports other than 91, as governed by the `cluster_admin` bind port set in the Gateway configuration. This change allows use of binding needed for the environment, such as for a proxy or in a Docker environment. (UIS-934)
- Fixed: Storage UI did not display search feed configuration if the Elasticsearch service was paused. (UIS-766)

Watch Items and Known Issues

- The UI reselects all nodes if deselecting them from the queue but do not launch the Restart when editing the Rolling Restart Queue. (UIS-957)
- A node may become temporarily unresponsive when attempting to view **Health Data** (the raw JSON of the health report) on the Advanced tab of the Chassis Details page. (SWAR-8349)
- A storage node may be reported to be in an unknown state rather than in maintenance mode while a reboot is in progress. (SWAR-8348)
- On the Details tab of the Chassis Details page, the Actions menu for each disk erroneously shows a testing-only command to **Fail** the disk. Selecting the command causes errors to display but does not affect the disk (UIS-955)
- Issues exist with feeds defined to use a non-default admin password. (UIS-759)

- Turn off the identify function before removing the disk from the chassis when using the Swarm UI to identify volumes. Failure to perform this can result in the need to restart the chassis. (UIS-564)
- Clicking **Add Export** on the NFS page causes an immediate 500 error if a cluster has no search feed defined. (UIS-441)
- The UI does not convey pausing cannot begin until the feed backlog is cleared when pausing a feed, which can be a long delay. (UIS-437)

Storage UI 3 Release

- [Changes in Storage UI 3.3](#)
- [Changes in Storage UI 3.2](#)
- [Changes in Storage UI 3.1](#)
- [Changes in Storage UI 3.0](#)

Changes in Storage UI 3.3

- Added configuration and safety guidance when deleting a primary search feed
- Updated chassis details page to include chassis serial number for easier cross-referencing with log messages
- Third party software package updates

Upgrade Impacts

Version Requirements:

- Swarm Storage 12.0.1 or higher
- Gateway 7.4

Changes in Storage UI 3.2

This release contains improvements to token handling, which affects SAML logouts and logins with expired tokens. (UIS-1093, UIS-1092, UIS-1078)

Watch Items and Issues – Same as 3.0.

Changes in Storage UI 3.1

Updating to this version is recommended to resolve potential issues with feed display and management and with moving between the Storage and Content UIs. (UIS-1084, UIS-1083, UIS-1081, UIS-1079, UIS-1076)

Watch Items and Issues – Same as 3.0.

Changes in Storage UI 3.0

Integrations

This version requires Gateway version 6.2 or later to use S3 Feeds, and SwarmFS version 3.0 or later, if used.

Single Sign-on – The Swarm Storage UI can now offer SSO (single sign-on) for your users through the new SAML 2.0 support in Content Gateway. The login page detects any SAML configuration for the requested host, tenant, or domain and redirects the user to log in with your identity provider, such as OneLogin, Okta, or Google. You can implement single sign-on globally, through the root IDSYS, and/or for specific tenants and domains, through the Content UI. See [Enabling SSO with SAML](#). (UIS-1072)



Troubleshooting 'Data Unavailable' – The Swarm UI now provides targeted errors and troubleshooting guidance for the range of issues that can prevent data from populating the charts for Swarm historical metrics ("Data unavailable"). (UIS-494)

NFS Exports – Dependency on Elasticsearch is removed for NFS export definitions. The Swarm UI **Settings > NFS, Edit export** page is updated to reflect the less complex NFS definitions. See [SwarmFS Export Configuration](#).

Watch Items and Issues

These are current operational limitations:

- The cache must be cleared to retrieve the current version after upgrading the Content UI (Portal) and/or Storage UI. Either shift-Reload the page or clear the browser cache, then verify the **About** page shows the new version. (UIC-222)

These are known issues:

- On the Chassis Details page, Advanced tab, the data for Health Data and View the raw JSON may not display. (UIS-1048)
- When using the Swarm UI to identify volumes, turn off the identify function *before* removing the disk from the chassis. Failure to perform this can result in the need to restart the chassis. (UIS-564)
- When a feed is paused, the UI does not convey that the pausing cannot begin until the feed backlog is cleared, which can be a long delay. (UIS-437)

Swarm Platform Release Notes

 **Note**

Review the changes and upgrade impacts for *each version* since the currently upgrading version if upgrading from a prior version.

- [Swarm Platform 14.1 Release Notes](#)
- [Swarm Platform 14.0 Release Notes](#)

Swarm Platform 14.1 Release Notes

- [Changes in Platform 14.1](#)
- [Limitations](#)

Changes in Platform 14.1

This release features the following improvements:

- **UI-based installation** - The SCS 2.0 is a UI-based solution for automatic installation and configuring the Swarm ecosystem on VMware vCenter.
- **Simplified deployment** - No complex CLI commands or steps are required as SCS 2.0 provides an ability to generate configurations based on the expected workload.
- **Simplified data protection solution** - Bucket lifecycle policies allow a bucket owner to use versions without worrying about the manual cleanup of old versions. This data protection mechanism mitigates concerns about runaway cluster space usage by limiting version accumulation.

 **Mandatory**

- vCenter credentials, vCenter infrastructure details, and Swarm Installer package is required to install and deploy SCS 2.0.

Limitations

- SCS 2.0 UI-based installation is performed in the VMware vCenter environment.
- Bucket Lifecycle Policy installation supports Gateway 7.9, Content Portal 7.6, and Swarm Storage 14.1 versions.

Swarm Platform 14.0 Release Notes

- [Changes in Platform 14.0](#)
- [Limitations](#)

Changes in Platform 14.0

This release features the following improvements:

- **Simplified cluster configuration** - Cluster configuration now happens via a centralized API. This eliminates the need to maintain multiple files at the cluster and node levels.
- **Improved command-line interface** - A simplified command-line interface now allows for easier Swarm administration. See [\[DRAFT\] Platform CLI Commands](#).
- **Simplified storage node deployment** - No per-node deployment or provisioning steps are required to start a Swarm Storage node. New nodes join the cluster on boot.

 **Required**

This version requires Swarm 14.0 or later, Gateway version 7.6 or later, and SwarmNFS version 2.1 or later, if used.

Limitations

- There is no UI support for this release (UIS-1137)
- There is no support for using gPXE instead of iPXE in this release (PLT-60)
- No offline mode installation in this release

Swarm Development

- [S3 Protocol Interface](#)
 - [S3 Object Locking](#)
 - [S3 Protocol Architecture](#)
 - [S3 Protocol Special Topics](#)
 - [Supported Amazon S3 Features](#)
 - [S3 Application Integration](#)
 - [S3 Protocol Configuration](#)
- [Storage SCSP Development](#)
 - [Development Guidance](#)
 - [Multipart Write](#)
 - [SCSP Essentials](#)
 - [Search Queries](#)
 - [Connecting to a Swarm Cluster](#)
 - [SCSP Methods](#)
 - [Metadata Headers](#)
 - [Content Integrity](#)
- [SCS CLI Commands](#)
- [Content Application Development](#)
 - [Gateway Metadata Transformation](#)
 - [Migrating Applications from Direct-to-Swarm](#)
 - [Metadata Translation between SCSP and S3](#)
 - [Token-Based Authentication](#)
 - [Restricting Domain Access](#)
 - [Gateway Audit Logging](#)
 - [Content Management API](#)
 - [Content SCSP Extensions](#)
- [Swarm SDK](#)
 - [SDK for C++](#)
 - [SDK Overview](#)
 - [SDK for C#](#)
 - [SDK for Python](#)
 - [SDK for Java](#)

S3 Protocol Interface

This section covers the software configuration of the S3 object storage protocol and provides guidance for integrating existing AWS S3 applications. Information in this document builds upon [Content Gateway Implementation](#) and [Content Application Development](#).

- [S3 Object Locking](#)
- [S3 Protocol Architecture](#)
- [S3 Protocol Special Topics](#)
- [Supported Amazon S3 Features](#)
- [S3 Application Integration](#)
- [S3 Protocol Configuration](#)

S3 Object Locking

- [Object locking does not prevent overwriting or deleting objects](#)
- [Retention](#)
 - [Retention Periods](#)
 - [How to Extend a Retention Period](#)
 - [Retention Modes](#)
 - [Retention mode applies to individual objects](#)
 - [Compliance mode is irreversible](#)
- [Legal Hold](#)
 - [Legal hold does not affect retention](#)
- [Prerequisites](#)
 - [Applications can impose user defined lifepoints together with object locks](#)
 - [Administrators are advised against disabling versioning once object locking is enabled anywhere in the cluster](#)
- [Metadata Headers related to Object Locking](#)
 - [Assumptions and Limitations](#)
 - [Lifepoint Headers](#)
- [How to Enable Object Locking](#)
 - [Enabling Object Locking on a Bucket](#)
 - [Object Locking and Versioning Inheritance Rules](#)
 - [Errors When Attempting to Enable Object Locking](#)
 - [Enabling Object Locking using S3](#)
 - [Enabling Object Locking using SCSP](#)
 - [Object Locking Cannot be Disabled After it is Enabled](#)
 - [Object locking cannot be disabled](#)
- [How to Check Object Locking Status](#)
- [REST API Changes](#)
- [How to Lock an Object at Creation Time](#)
 - [Creating a New Object with a Retention Period](#)
 - [Creating a New Object with a Legal Hold](#)
 - [Writing an Object as a Normal Unlocked Object in a Bucket with Object Locking Enabled](#)
- [Managing Retention on an Existing Object](#)
- [Managing Legal Hold on an Existing Object](#)
- [Combined Retention and Legal Hold](#)
- [Differences Between S3 and Swarm's Implementation of Object Locking](#)
- [New Policy Actions Related to Object Locking](#)
- [Interactions with Existing Swarm Functionality](#)
 - [Recursive Deletes](#)
 - [APPEND](#)
 - [Max Retention Configuration](#)
 - [Audit Logging](#)

Object locking prevents object versions from being deleted or overwritten – for a fixed amount of time or indefinitely. Use an object lock to help meet regulatory requirements requiring WORM storage, or to add another layer of protection against object changes and deletion.

Objects are not actually locked. Object locking is used to lock **individual object versions**.

Object locking does not prevent the creation of new versions of an object while the object is locked – it makes it impossible to delete or otherwise change the version(s) of the object with locking enabled.

Object locking does not prevent overwriting or deleting objects

Since the locked **versions** remain present and protected it is still possible to overwrite or even delete objects with locking enabled. A new version is created when overwriting or modifying an object. A delete request creates a delete marker. The object appears deleted, but Swarm preserves history including the locked version.

There are two types of object locking used simultaneously and independent of each other:

- **Retention** – Specifies a fixed period of time ("retention period") during which the object remains locked. During this period, the object is WORM-protected and cannot be overwritten or deleted. The lock goes away automatically after the period expires.
- **Legal hold** – An object stays locked indefinitely when a legal hold is applied. A legal hold does not expire; it must be explicitly removed.

Retention

Retention Periods

A **retention period** is used to set the fixed amount of time the object needs to remain locked. The object version cannot be changed or deleted until the time expires.

There is more than one way to set a retention period on an object version:

- Newly created objects can inherit a **default retention period** configured on the bucket level.
- Explicitly setting a retention period when creating a new object overrides the default retention period configured for the bucket if present.
- Explicitly set a retention period on an existing object version.

A **bucket default retention period** specifies a duration (in days or years) for which every object version placed in the bucket is locked. Gateway calculates a retention period for the object version by adding the specified duration to the object version's creation timestamp when placing an object in the bucket.

How to Extend a Retention Period

A retention period can always be extended after it is set with the following steps:

1. Submit a new lock request for the object version with a retention period longer than the current one.
2. Gateway replaces the existing retention period with the new, longer period.

Any user with permissions to set an object retention period can also extend a retention period.

Retention Modes

A **retention mode** must always be specified when locking an object or setting a bucket default retention period.



Retention mode applies to individual objects

Retention mode always applies to the individual objects carrying it, not to the bucket or cluster as a whole.

There are two retention modes impacting actions with objects under retention:

- In **governance mode**, grant some users the permission to shorten or remove a retention period or delete object versions under retention if necessary.
- In **compliance mode**, a locked object version cannot be overwritten or deleted by any user, even the admin user. The retention mode cannot be changed, and the retention period cannot be shortened when an object is locked in compliance mode.

i Compliance mode is irreversible

Compliance mode is irreversible for the entire retention period once an object is locked.

In a deviation from S3, Gateway always uses the maximum of either the bucket default retention duration, or the duration specified in a per-object request.

Legal Hold

A legal hold prevents an object version from being overwritten or deleted until the legal hold is removed.

Legal holds do not have an associated retention period and are independent from retention periods and retention modes. As long as the bucket containing the object has object locking enabled adding and removing legal holds are available. It does not matter if the specified object version has a retention period set or not.

i Legal hold does not affect retention

Setting a legal hold on an object version does not affect the retention mode or retention period for the object version.

Prerequisites

Swarm Storage 12.0 or above must be running to use this feature because it relies on the Swarm [lifepoints](#) feature to prevent deletion of locked objects until a certain date has passed. Object locking is fully implemented starting with Gateway 7.6.

When an object is locked until a certain date, it obtains a `deletable=no` lifepoint protecting it from deletion until the date.

i Applications can impose user defined lifepoints together with object locks

Even though Gateway relies on lifepoints, it remains possible for applications to impose user defined lifepoints on objects together with object locks. Gateway verifies correct semantics in all cases without any additional behavior needed from the application side. In case of any conflicts between user defined lifepoints and object locks, the object lock always wins.

[Versioning](#) needs to be enabled for object locking to work.

Gateway refuses to enable object locking when versioning is not enabled. Gateway refuses to disable versioning once object locking is enabled. In both cases an error message is displayed.

Administrators are advised against disabling versioning once object locking is enabled anywhere in the cluster

The ability to disable versioning at the cluster level via SNMP does not pass via Gateway so it cannot protect against disabling object locking in the cluster.

Administrators are advised against disabling versioning at the cluster level to avoid the risk of auto-deleting locked object versions after object locking is enabled in individual domains or buckets.

Metadata Headers related to Object Locking

Object versions in Swarm are immutable and the metadata cannot be changed unlike Amazon S3.

Object locking uses the following headers:

- on **buckets**:
 - `x-object-lock-meta-status: ENABLED`
(empty means *DISABLED*)
 - `x-object-lock-meta-default: <GOVERNANCE | COMPLIANCE>[:<duration>]`
Bucket default retention period duration is expressed as `<integer>y` for number of years or `<integer>d` for number of days.
- on **objects**:
 - `x-object-lock-meta-mode: <GOVERNANCE | COMPLIANCE>`
 - `x-object-lock-meta-retain-until-date: <date>`
 - `x-object-lock-meta-legal-hold: ON`
(empty means *OFF*)
 - `x-object-lock-meta-original-lifepoints:`
`<original lifepoints>`
 - `lifepoint: [<date>] deletable=no`
(for retention period)
 - `lifepoint: [] deletable=no`
(for legal hold)

The above headers are listed using the SCSP names. The corresponding S3 names start with `x-amz-*`. The SCSP headers are effectively stored with the objects. The S3 names are mapped onto the SCSP counterparts and back on-the-fly.

Assumptions and Limitations

- Internal to Gateway, all header values are treated as case-insensitive.
- Dates are in the `rfc1123` format, e.g. "Wed, 12 Dec 2016 15:59:02 GMT". For S3 these are translated in to the `ISO8601` format.
- The `x-object-lock-meta-retain-until-date` header applies to retention periods and specifies the end date of the retention period. The `x-object-lock-meta-legal-hold` header applies to legal hold. Both a retention period and a legal hold can both be set on the same object version.

- The `x-object-lock-meta-original-lifepoints` header stores the complete set of user defined delete/deletable lifepoint headers found on the object at the time the retention period/legal hold was applied. The original delete/deletable lifepoint headers are removed. Swarm no longer considers these lifepoints. Gateway manipulates delete/deletable type lifepoints, all other lifepoints are unaffected and continue functioning normally. An object lock in effect takes precedence over any user-defined delete/deletable lifepoints, blocking `delete`. The user-defined lifepoints take effect again when an object lock expires or is removed.
- Object locking works for tenanted objects. Object locking cannot be set on untenanted and unnamed objects in a cluster.

Lifepoint Headers

Gateway now adds a single `deletable=no` lifepoint header (the **lock lifepoint**), to go along with the object lock. This lock lifepoint is what actually protects the object against deletion in Swarm, both through user requests and through built-in functionalities like HP or bucket policies.

The lock lifepoint is computed as follows;

1. The lock lifepoint end date matches the end date of the retention period if the object is locked with a retention period. For legal hold, the lock lifepoint has no end date.
2. Next review the list of original lifepoints, and append those whose end date is later than the one from the lock lifepoint. In case of legal hold, there is no end date so none of the original lifepoints get appended.

The purpose of storing the set of original lifepoints is to allow later modifications/removal of the object lock to recompute/reinstate the original lifepoints as they are before the object lock.

The purpose of appending the "later lifepoints" to the lock lifepoint is to allow Swarm to act on them as it normally does once the lock lifepoint has expired naturally, without any intervention from Gateway. For legal hold there must always be a Gateway intervention to remove the lock, so the original lifepoints get reinstated at the time.

How to Enable Object Locking

Object locking is set using API calls. It is not available in the user interface.

Enabling Object Locking on a Bucket

Object locking must be enabled on the bucket before locking any objects.

S3 normally allows enabling object locking on new buckets without any objects. Gateway does not impose this restriction.

The user must have the `PutBucketObjectLocking` permission to enable/disable object locking on a bucket. A user must have the `GetBucketObjectLocking` permission to query current object locking status.

Object Locking and Versioning Inheritance Rules

Versioning can be disabled for another bucket or domain in the cluster unrelated if a bucket in one domain has object locking (and therefore versioning) enabled.

Cannot be disabled at the bucket, domain or cluster level if object locking (and versioning) is enabled at a bucket level.

Can be disabled for individual buckets if enabled at a domain level.

Can be disabled for individual domains and/or buckets if enabled at the cluster level.

It does not matter if versioning was enabled on the bucket itself, or whether it was inherited from cluster or domain level. Gateway refuses to disable versioning at the domain or bucket level if object locking is in effect anywhere within it.

Errors When Attempting to Enable Object Locking

The request to enable object locking can fail with the following errors:

- `412 Precondition Failed` is displayed if the Swarm cluster does not support all features necessary to perform the operation.
- `412 Precondition Failed` is displayed if the bucket does not have versioning enabled.
- `403 Forbidden` if the user does not have the `PutBucketObjectLocking` permission.

Enabling object locking on a bucket comes down to storing the `x-object-lock-meta-status` and optionally `x-object-lock-meta-default` headers on the bucket context object. This immediately caches the bucket's object locking configuration in memory so it is readily available during object requests.

Enabling Object Locking using S3

Enable or inspect the object locking configuration on a bucket using the following calls;

- https://docs.aws.amazon.com/AmazonS3/latest/API/API_PutObjectLockConfiguration.html
- https://docs.aws.amazon.com/AmazonS3/latest/API/API_GetObjectLockConfiguration.html

Enabling Object Locking using SCSP

Enable object locking on a bucket:

```
PUT /<bucket>?objectlock=<defaultmode> [ :<defaultperiod> ]
```

- `defaultmode` can be either "governance" or "compliance"
- `defaultperiod` is optional; it is a number of years (y) or days (d), eg. 1y or 20d.

In this call omit either `defaultmode`, `defaultperiod`, or both.

The bucket allows object locking (it is enabled), but no locking happens by default if both are omitted. Objects written to the bucket without any locking directives remain unlocked.

The defaults can be modified or removed at any time via additional `PUT` commands. This does not affect the object locking status of the bucket – once it is enabled, it stays enabled.

Object Locking Cannot be Disabled After it is Enabled



Object locking cannot be disabled

Object locking cannot be disabled once enabled on a bucket.

It does not matter if versioning was enabled on the bucket itself, or whether it was inherited from cluster or domain level. Gateway refuses to disable versioning at the domain or bucket level if object locking is in effect anywhere within it.

Remove the lock defaults and write new objects without any object locking headers to allow writing unlocked objects into a bucket with object locking enabled.

Remove the lock defaults:

```
PUT /<bucket>?objectlock=
(use an empty query argument)
The "=" is optional.
```

How to Check Object Locking Status

- Query the object locking status of a bucket:

```
GET /<bucket>?objectlock
```

This returns the following response headers:

```
x-object-lock-meta-status: ENABLED
```

```
x-object-lock-meta-default: <GOVERNANCE|COMPLIANCE>[:<duration>]
```

And the response body says:

```
Object locking is enabled on bucket <bucket> with default mode <mode> [ and default
duration <duration> ]
```

No response headers are present and the response body displays as below if the bucket does not have object locking enabled:

```
Object locking disabled
```

REST API Changes

Object locking introduces the following new REST calls:

- **PUT** with `?objectlock` query arguments
- **GET** with `?objectlock` query arguments
- **DELETE** with `?objectlock` query arguments (**DELETE** is an object-only call)

Object locking also introduces changes to existing REST calls:

- **PUT** can now take object locking headers to create locks as a side effect
- **POST** can now take object locking headers to create locks as a side effect
- **COPY** can now take object locking headers to create locks as a side effect
- **GET** can return object locking headers if the user has the appropriate permissions
- **HEAD** can return object locking headers if the user has the appropriate permissions

For S3, see [the Amazon documentation website](#).

How to Lock an Object at Creation Time

Creating a New Object with a Retention Period

The client adds the following headers in the S3 `PutObject` / SCSP `POST` request to create a new object or object version with an immediate retention period in effect:

```
x-object-lock-meta-mode: <GOVERNANCE | COMPLIANCE>
x-object-lock-meta-retain-until-date: <date>
```

This takes precedence over the default bucket retention mode and duration, if present.

In a deviation from S3, Gateway always uses the maximum of either the bucket default retention duration, or the duration specified in a per-object request.

Gateway looks for corresponding defaults at the bucket level if any one of these two headers is omitted from the request and if found, takes the corresponding values from there.

The request fails with a `400 Bad Request` error because both are needed for a successful retention lock if either mode or retain-until-date is then still missing.

Creating a New Object with a Legal Hold

The client adds the following header in the S3 `PutObject` / `SCSP POST` request to create a new object or object version with an immediate legal hold in effect:

```
x-object-lock-legal-hold: ON
```

The user needs to have the `PutObjectRetention` and, respectively, `PutObjectLegalHold` permission to use these headers, or the request fails with a `403 Forbidden` error.

The request fails with a `412 Precondition Failed` error if the bucket does not have object locking enabled.

Gateway forwards these headers when creating the new object on Swarm, and also creates a lock lifepoint instructing Swarm to not delete the object before the retention period expires. For retention periods, the lock lifepoint also includes the subset of the original lifepoints with a later end date than the retention period.

```
lifepoint: [<date>] deletable=no, <later lifepoints>
```

Or in case of legal hold;

```
lifepoint: [] deletable=no
```

The original lifepoint headers are preserved in:

```
x-object-lock-meta-original-lifepoints: <original lifepoints>
```

Writing an Object as a Normal Unlocked Object in a Bucket with Object Locking Enabled

Objects are written as a normal unlocked object, despite being written to a bucket with object locking enabled if both headers are omitted from the request, and there are no defaults at the bucket level.

Managing Retention on an Existing Object

Enabling and disabling retention on an object requires the user have the `PutObjectRetention` permission.

The user must have the `GetObjectRetention` permission to query current retention status.

The client must explicitly specify the `versionId` of the object version to lock.

Gateway then applies the extra object locking headers to the version, thus applying object locking protection for the length of the retention period.

The following headers are added or changed;

```
x-object-lock-meta-mode: <GOVERNANCE | COMPLIANCE>
x-object-lock-meta-retain-until-date: <date>
x-object-lock-meta-original-lifepoints: <original lifepoints>
lifepoint: [<date>] deletable=no, <later lifepoints>
```

Gateway must recompute the lock lifepoint when changing the retention period, starting from the preserved set of original lifepoints and appending those with later end dates to the lock lifepoint. This is the main purpose of preserving the original lifepoints.

In a deviation from S3, Gateway always uses the maximum of either the bucket default retention duration, or the duration specified in a per-object request.

Introducing or extending a retention period is always possible, but there are restrictions to shortening or removing a retention period on an object already under retention:

- In **compliance mode** this is not permitted
- In **governance mode**, the user needs to have the special `BypassGovernanceRetention` permission. Also, an S3 request must explicitly include `x-amz-bypass-governance-retention:true` as a request header with any request requiring overriding governance mode.

Using S3, enable or inspect the retention period on an object using the following calls;

- https://docs.aws.amazon.com/AmazonS3/latest/API/API_PutObjectRetention.html
- https://docs.aws.amazon.com/AmazonS3/latest/API/API_GetObjectRetention.html

Using SCSP, enable or inspect the retention period on an object using the following calls:

- Set a governance lock onto an object, specifying both lock mode and duration (this overrides any defaults configured on the bucket):
`PUT /<bucket>/<object>?version=<uuid>&objectlock=governance:<untildate>`
- Set a compliance lock onto an object and inherit the default duration from the bucket:
`PUT /<bucket>/<object>?version=<uuid>&objectlock=compliance`
- Completely inherit the default object lock mode and duration on the bucket:
`PUT /<bucket>/<object>?version=<uuid>&objectlock`
- Remove a (governance) object lock from an object:
`DELETE /<bucket>/<object>?version=<uuid>&objectlock=<mode>`
The mode is always "governance". A compliance mode object lock cannot be removed. The `BypassGovernanceRetention` permission is required to carry out this action and the request must carry the `x-object-lock-meta-bypass-governance:true` header.
- Query current object lock status:
`GET /<bucket>/<object>?version=<uuid>&objectlock`
The response carries the following headers:
`x-object-lock-meta-mode: <GOVERNANCE | COMPLIANCE>`
`x-object-lock-meta-retain-until: <date>`
And the response body says:
Object is locked in <mode> mode until <date>
None of the headers are present when called on an object not under retention, and the response body says:
Object is not locked

Both S3 and SCSP also allow retrieving object lock information using regular object `HEAD` and `GET` requests.

Assuming the user has the `GetObjectRetention` permission, the information is returned in the form of the above response headers. The response body is not affected.

Managing Legal Hold on an Existing Object

Enabling/disabling legal hold requires the user have the `PutObjectLegalHold` permission. The `GetObjectLegalHold` permission is required to check the current legal hold status.

As with retention periods, Gateway stores this as a metadata header.

```
x-object-lock-meta-legal-hold: ON
(empty means OFF)
x-object-lock-meta-original-lifepoints: <original lifepoints>
lifepoint: [] deletable=no
```

Using S3, enable or inspect the legal hold using the following calls:

- https://docs.aws.amazon.com/AmazonS3/latest/API/API_PutObjectLegalHold.html
- https://docs.aws.amazon.com/AmazonS3/latest/API/API_GetObjectLegalHold.html

Using SCSP, enable or inspect the legal hold on an object using the following calls:

- Set a legal hold onto an object:
`PUT /<bucket>/<object>?version=<uuid>&objectlock=legal-hold`
- Remove a legal hold from an object:
`DELETE /<bucket>/<object>?version=<uuid>&objectlock=legal-hold`
This reinstates any original lifepoints by moving them from the `x-object-lock-meta-original-lifepoints` header back in to the proper `lifepoint` headers.
- Query an object's legal hold status:
`GET /<bucket>/<object>?version=<uuid>&objectlock`
The response carries the following headers:
`x-object-lock-meta-legal-hold: on`
And the response body says:
`Object is locked in legal hold`
The header is not present when called on an object not under legal hold nor retention, and the response body says:
`Object is not locked`

Both S3 and SCSP also allow retrieving legal hold information using regular object `HEAD` and `GET` requests. Assuming the user has the `GetObjectLegalHold` permission, the information is returned in the form of the above response headers. The response body is not affected.

Combined Retention and Legal Hold

An object can be both under one of the retention modes AND legal hold at the same time.

In the SCSP protocol, querying and deleting such combined locks is handled via a uniform `GET` and `DELETE` API (as opposed to S3 which has separate APIs for querying/deleting retention and legal hold).

- Query:
`GET /<bucket>/<object>?version=<uuid>&objectlock[=<locktype>]`
Response headers for both retention and the legal hold display when querying the object lock status without specifying the lock type, and the response body contains both status texts, separated by a new line.
The user needs both the `GetObjectRetention` and the `GetObjectLegalHold` permissions for this request.
Query the lock status for one specific lock type, either "legal-hold" or "retention". The corresponding permission is required.

- **Delete:**
`DELETE /<bucket>/<object>?version=<uuid>&objectlock=<locktype>`
 Using SCSP remove either the retention or legal hold using DELETE and specifying the appropriate query argument `objectlock=<locktype>`, locktype being "legal-hold" or "retention". The locktype of "retention" is used exclusively for locks in governance mode.

Differences Between S3 and Swarm's Implementation of Object Locking

In S3, a DELETE request results in a delete marker, shadowing the locked object version. Swarm's implementation deviates from this logic – it rejects any DELETE requests for indelible objects with an HTTP 403 Forbidden error.

Gateway checks if the object is locked when it receives an HTTP 403 Forbidden error from Swarm. Gateway simulates the S3 behavior creating a new (unlocked) object version, immediately followed by a DELETE, thus creating a delete marker.

For SCSP, use a configuration flag to pick the desired behavior:

- fail deletes of locked objects with a 403 Forbidden error, or
- mimic [the S3 behavior](#)

```
[object_locking]
scspDeleteUsesS3Logic=true
```

New Policy Actions Related to Object Locking

The following new policy actions related to object locking are introduced:

- `PutBucketObjectLocking`: to enable/disable object locking on a bucket
- `GetBucketObjectLocking`: to query bucket object locking status
- `PutObjectRetention`: to set or extend object retention
- `GetObjectRetention`: to query object retention
- `BypassGovernanceRetention`: to shorten/remove a retention in governance mode
- `PutObjectLegalHold`: to set/remove a legal hold
- `GetObjectLegalHold`: to query legal hold

Interactions with Existing Swarm Functionality

Recursive Deletes

Clients can request [recursive deletes](#) of entire domains/buckets using `DELETE <uri>?recursive=yes` requests. Swarm implements this by synchronously deleting the domain/bucket object, and asynchronously deleting the objects in it. This has potential for conflicting with object retention/legal hold. Gateway first checks if there are any objects under retention/legal hold and refuse the recursive delete if so.

The recursive delete request fails with a 412 Precondition Failed error if any buckets with object locks are found.

APPEND

[SCSP APPEND](#) does create a new version when versioning is enabled; it does not allow an object's metadata to be modified. The "retain-until" header value can not be changed on an `APPEND`. Gateway does not impose any restrictions on the use of `APPEND` in combination with object locking.

Max Retention Configuration

S3 allows defining a `max-retention-duration` limit in the policy. Gateway currently approximates this functionality using a new configuration flag:

```
[object_locking]
retentionMaxYears=100
The default limit value is 100 years (input type int) if unspecified.
```

- A year is assumed to be 365 days when performing conversions between numbers of days and years.
- In the SCSP/S3 APIs, any user-specified value exceeding the limit is capped to the limit.

Audit Logging

Object Locking operations are [audit logged](#). Since object locks can also be requested as part of the object `PUT/POST/COPY` requests, Gateway tag the request's audit log line with additional object lock information, rather than inserting new log lines.

The tags are appended to the audit log line, enclosed in `[]` brackets. Multiple tags (for example, both legal hold and retention are requested) are comma separated.

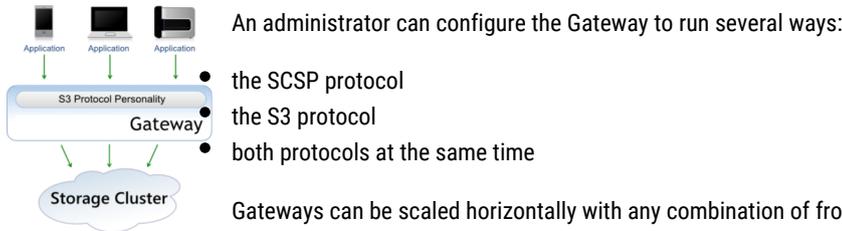
Object locking tags are always prefixed with `OBJLCK`.

- Enabling retention on a bucket and setting defaults if provided:
`<audit log line> [OBJLCK:ENABLE:<mode>:<duration>]`
- Setting/removing retention on an object:
`<audit log line> [OBJLCK:RETENTION:<mode>:<retainUntil>]`
`<audit log line> [OBJLCK:RETENTION:NONE]`
- Setting/removing legal hold on an object:
`<audit log line> [OBJLCK:LEGALHOLD:ON]`
`<audit log line> [OBJLCK:LEGALHOLD:OFF]`

S3 Protocol Architecture

- [Sharing Storage across S3 and SCSP](#)
- [Routing Methods](#)

The S3 protocol personality is a front-end storage protocol for client applications. It runs within the Gateway itself. All Gateway deployment and scaling features apply when using the S3 front-end protocol.



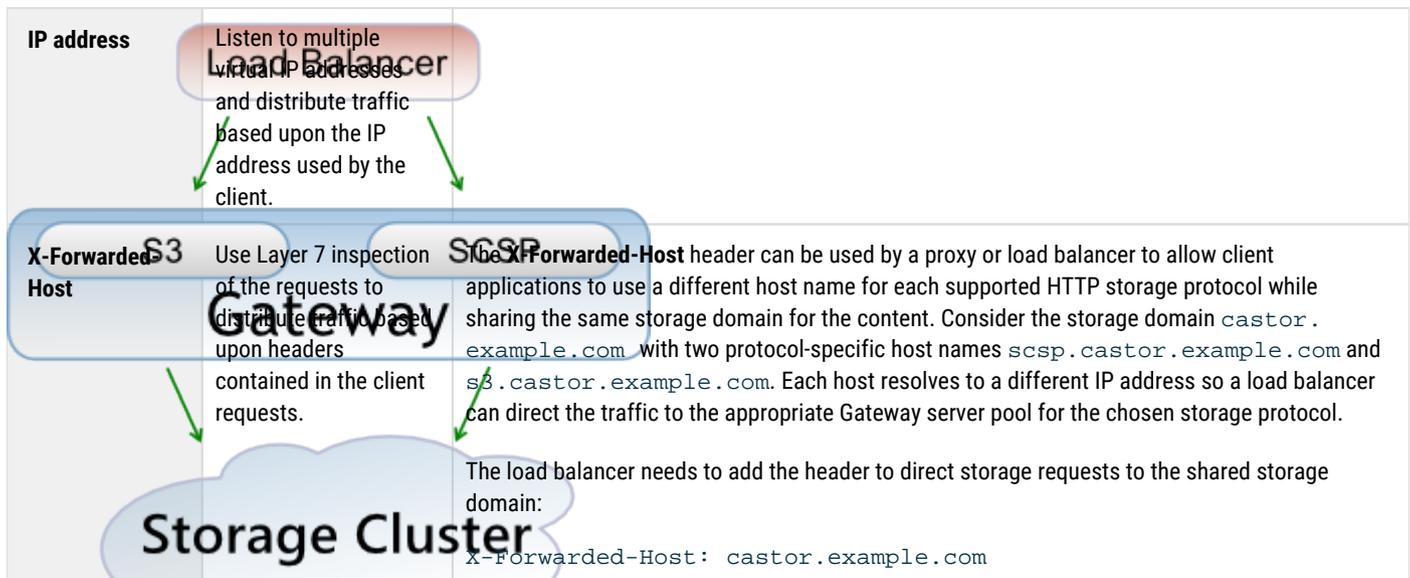
Gateways can be scaled horizontally with any combination of front-end protocols as needed for a particular deployment. This allows for a heterogeneous mix of client types utilizing the same Swarm storage cluster and sharing content with each other. Performing this allows an administrator to provide a unified object storage platform.

Sharing Storage across S3 and SCSP

Content Gateway provides the mechanism to unify the back-end Swarm object storage so S3 applications and SCSP applications can share content. In a unified object storage deployment, a device like a load balancer is used to route incoming client traffic through the appropriate port number or pool of Gateway servers.

Routing Methods

These are some routing methods usable for a unified object storage front-end:



DNS	Cause the DNS name resolution to be different for the clients using one storage protocol than for the clients using another protocol.	Clients using S3 may resolve the storage domain <code>castor.example.com</code> to <code>10.100.100.81</code> while clients using SCSP resolve the same storage domain to <code>10.100.100.82</code> . This method requires the administrator has control of the hosts where the client applications run so as to allow alteration of the DNS/host resolution but does not require in-line modification of the HTTP requests.
OPTIONS	Route request method OPTIONS with "Origin" to the S3 port rather than default to the SCSP port (which happens when an Authorization header does not exist or have "AWS").	S3 must handle "bucket in Host" style requests because SCSP reports the domain was not found. <pre>curl -i -H 'Origin: http://www.example.com' -H 'Access-Control-Request-Method: PUT' -X OPTIONS https://mycorsbucket.elsewhere.com/</pre>
Pattern-matching	Use Layer 7 inspection of the requests to switch incoming traffic based on the object storage protocol. See below:	This is performed by looking for the distinctive S3 Authorization header pattern or one of the query string arguments for authenticated S3 requests. The pattern-matching rules for these authenticated S3 requests are as follows.
<ul style="list-style-type: none"> Headers 	Use pattern-matching on the request header.	<pre>Authorization =~ ^AWS.*</pre>
<ul style="list-style-type: none"> Arguments 	Use pattern-matching on the query string arguments.	<pre>{RequestURL} =~ [?&](AWSAccessKeyId X-Amz-Credential)=</pre>
<ul style="list-style-type: none"> Absence 	Use pattern-matching to test for the <i>absence</i> of all AWS request patterns.	The request is either an anonymous S3 request or an SCSP request. Since anonymous S3 requests do not create, update, or delete content, they are most likely GET requests, and it is safe to allow SCSP to handle these.

S3 Protocol Special Topics

Amazon S3 is two distinct things: "S3 The Service" and "S3 The Protocol." Since the S3 protocol reflects characteristics of the Amazon service that may not be applicable outside of Amazon, the Gateway adapts these characteristics so they make sense when hosting storage within the environment. These adaptations are transparent to most applications and enhance the protocol features by making use of the unique strengths of Swarm storage.

- [AWS Regions versus Storage Domains](#)
 - [Best practice](#)
- [Bucket Location](#)
 - [Best practice](#)
- [Storage Class](#)
- [Virtual Hosting of Buckets](#)
- [Identity Management and S3 Authentication Tokens](#)
 - [Note](#)
- [Error Response for Missing Resource](#)
- [Multipart Uploads](#)
- [List after Update Timing](#)
- [PUT Object Copy Metadata](#)
- [S3 Versioning](#)

AWS Regions versus Storage Domains

Amazon AWS currently has fewer than [two dozen geographic regions](#) worldwide, each of which is shared by thousands of end-users. Choose a permanent region for each of the S3 buckets based upon latency, cost, and any applicable regulatory requirements.

An Amazon AWS region is roughly analogous to a Swarm storage domain. A Swarm bucket is tied to a domain as an S3 bucket is tied to a region. They are fundamentally different in the following ways:

- AWS regions are finite, but any number of Swarm domains can be created; this allows achieving the optimal granularity for assigning content user permissions.
- AWS buckets must have a name unique across *all* regions, but Swarm buckets need only be unique within the domain.
- AWS buckets are fixed geographically, but Swarm content dynamically distributes across the entire Swarm cluster, according to the content protection settings.
- AWS buckets can be replicated to a differently named bucket, but Swarm replication preserves the domain, bucket, and object names identically across every target cluster, which supports content sharing and direct DR fail-over.
- AWS buckets can replicate in one direction, but Swarm clusters can use mirrored replication. A domain's content can be created, updated, and accessed across distributed storage clusters while remaining synchronized.



Best practice

Domain creation and allocation in Swarm is lightweight, so be generous: provide each client (or business unit) a separate storage domain. Performing this benefits both sides:

- For storage admins, usage tracking and storage management are easier.
- For storage end-users, access control policies are uncomplicated, and bucket naming is more flexible.

Bucket Location

The Amazon S3 *GET Bucket Location* request returns the AWS region in which the bucket is located. This request in the Gateway's S3 protocol implementation returns the value of the `cluster.name` parameter configured in Swarm. The return value for a bucket location request depends upon the cluster serving the request if a storage domain exists in more than one cluster.

Unlike Amazon S3 where the geographic location of a bucket is chosen when it is created and stays fixed, Swarm cluster placement for a storage domain and the buckets it contains is controlled by the storage administrator. Additionally, a domain may exist in multiple storage clusters if the administrator has setup remote replication.

Best practice

Provide every cluster a unique cluster name so applications can use the name to identify the location where the content is being served.

Storage Class

Amazon S3 allows clients to set a storage class preference in the `x-amz-storage-class` header, which defines the data durability and access frequency of content. See the [Amazon S3 Storage Classes](#).

The S3 protocol tags *all* objects with the `x-amz-storage-class-meta` header and includes the client application's requested class or STANDARD, if none is specified. Bi-directional translation between the AWS S3 `x-amz-storage-class` header and the Swarm `x-amz-storage-class-meta` header is performed for S3 protocol operations.

Virtual Hosting of Buckets

Amazon S3 allows virtual host name to bucket mappings within the storage service. This is accomplished by creating a DNS alias (CNAME record) for a virtual host name pointing to one of the Amazon S3 region end-points. An example is to allow the web request to be mapped to the real Amazon S3 URL:

```
http://www.fred.com/hello.html
http://s3.amazonaws.com/www.fred.com/hello.html
```

The HTTP `Host` header can be used to specify the bucket name if the virtual host is mapped to the *bucket* named `www.fred.com` in the US Standard Region.

The S3 protocol also supports this mapping of virtual hosts to buckets. To accomplish this, the storage administrator configures the DNS server to perform wildcard resolution of host names to the front-end IP address of the Gateway. The Gateway then search for a storage domain within Swarm starting with the value of the `Host` header and then recursively popping off the leftmost word using period (".") as a delimiter until it finds a match or runs out of words. The previously popped words are concatenated back together using periods and the result becomes the bucket name for the request when a storage domain is found.

As an example, consider the storage domain in Swarm called `fred.com` containing a bucket called `www` and an object within called `hello.html`. The normal method to access this object is with the URL `http://fred.com/www/hello.html`, which has the following HTTP request headers:

```
GET /www/hello.html HTTP/1.1
Host: fred.com
```

Create DNS entries for `www.fred.com` and `fred.com` to setup the virtual host mapping from `www.fred.com` to the Swarm storage domain and bucket. This is an example where the Content Gateway's front-end IP address is `10.100.100.81` and a wildcard match is used.

```
fred.com      A      10.100.100.81
*.fred.com   CNAME  fred.com
```

After the DNS entries are in place, the `hello.html` object is now accessible with the additional URL `http://www.fred.com/hello.html`. The HTTP request headers arriving at the Gateway look like this:

```
GET /hello.html HTTP/1.1
Host: www.fred.com
```

The Gateway first checks for the non-existent storage domain `www.fred.com` and then removes `www`, the leftmost word, and finds the storage domain `fred.com` in the storage cluster. The Gateway then transparently modifies the HTTP request headers by prefixing the URI path with the removed word `www` and shortening the `Host` header as follows:

```
GET /www/hello.html HTTP/1.1
Host: fred.com
```

The Gateway searches for a storage domain by removing the leftmost words until a domain is found or until it reaches a null host name if the bucket name contains periods, `hires.images` with a virtual host name of `hires.images.fred.com`. For example, it tests for the existence of the domains `hires.images.fred.com` and `images.fred.com` before finding `fred.com`. The request results in an error if the search reaches a null host name. The `[caching] domainExistenceRefresh` configuration parameter in `gateway.cfg` is used to optimize domain existence testing.

Identity Management and S3 Authentication Tokens

The Gateway's S3 protocol uses an external identity management system (IDM) for users and groups, similar to federation with AWS IAM, and uses an internal system for managing authentication tokens, similar to AWS temporary security credentials. These authentication tokens are created and managed within the Swarm cluster.

Authenticated requests to the S3 protocol follow the AWS S3 request signing rules for v2 and v4 signatures whereby each request includes a header in one of the following forms:

- `Authorization: AWS AccessKey:Signature` (v2 signature)
- `Authorization: AWS4-HMAC-SHA256 Credential,SignedHeaders,Signature` (v4 signature)

The construction of the `Authorization` header is automatically handled by S3 SDKs and S3 applications. For the the elements of the header's value string and the S3 HMAC authentication mechanism, see the [AWS S3 documentation](#).

Each S3 client needs at least one authentication token to authenticate S3 protocol requests to the Gateway. The Access Key value is the ID of an authentication token.

See [Token-Based Authentication](#).

The `X-User-Secret-Key-Meta` header is required when creating the token object when creating authentication tokens for use with S3 (also referred to as "S3 authentication tokens"). The value of this header becomes the Secret Access Key (or "Secret Key") used to sign S3 requests. As previously mentioned, the Access Key becomes the token cookie's value returned by the create request.

This example shows an authentication token being created by the user `gcarlin` with an S3 secret key of `abcdefg` and an expiration time of 365 days from now. Note: this request uses HTTP basic authentication to create the token.

```
POST /.TOKEN/ HTTP/1.1
Authorization: Basic Z2NhcmxpbjpmYW5ueQ==
Host: abc.cloud.example.com
X-User-Secret-Key-Meta: abcdefg
X-User-Token-Expires-Meta: +365
Content-Length: 0
```

This is an excerpt from the Gateway's response.

HTTP/1.1 201 Created

Set-Cookie: token=722bfb49aa8365897a3e774d539038ce; expires=Fri, 06-Jun-2017 18:44:52 GMT; path=/

Construct the S3 Authorization header using the following to sign S3 requests with the created token:

AccessKey=722bfb49aa8365897a3e774d539038ce

SecretAccessKey=abcdefg



Note

Tokens that contain an S3 secret key are used to sign S3 storage requests and may not be used to authenticate SCSP storage operations.

Error Response for Missing Resource

The S3 protocol includes an extra XML `Resource` tag in the error response to provide additional details for troubleshooting errors regarding non-existent content. This is an example showing the additional field.

```
<Error>
  <Code>NoSuchKey</Code>
  <Message>The specified key does not exist.</Message>
  <Resource>/mybucket/missingFile.txt</Resource>
  <RequestId>03B8CD915CD6C3A5</RequestId>
  <Key>missingFile.txt</Key>
</Error>
```

The format of the `Resource` tag is: `"/{bucketName}/{objectName}"`.

Multipart Uploads

AWS S3 requires every part (except the last) of a multipart upload must be at least 5 MB in size. The Gateway's S3 protocol implementation does not impose this limitation and allows each part of a multipart upload to be of any size.

Multipart writes are long-running operations with initial and final responses. For S3 compatibility, the initial response returns `x-amz-version-id` with the value of the expected ETag. There is no new object if there is an error completing the write, and the expected ETag given is not valid. (v6.3)

List after Update Timing

After an object is created, the delay before that object appears in a list operation can vary depending upon the Swarm metadata feed batch timeout setting. New objects are available within two seconds following a create when the batch timeout is set to 1 second. The [Amazon S3 documentation](#) has specific developer guidance about this eventual consistency behavior.

PUT Object Copy Metadata

The AWS S3 *PUT Object Copy* request creates a duplicate of an existing object and, when the `x-amz-copy-source` header is included with the request, copies the `x-amz-meta-*` custom metadata from the source object.

Although Swarm allows for more custom metadata patterns than this, the metadata matching `x-amz-meta-*` pattern is copied during a *PUT Object Copy* operation.

S3 Versioning

Swarm's native object versioning feature is interoperable with AWS S3 versioning. The implementation includes these improvements:

- *Ability to disable versioning:*
AWS S3 allows for versioning to be suspended once enabled on a bucket. Swarm provides the ability to disable versioning and automatically clean up the prior versions to reclaim storage space.
- *Delete marker consolidation:*
Unlike AWS S3 where continued DELETE operations on a deleted object record additional delete markers in the version history, Swarm acknowledges the subsequent deletes without recording additional delete markers. Multi-factor authentication delete is not supported.
- *Expanded version listing:*
Swarm supports version listing batches up to 2000 items while AWS S3 limits these listing results to batches of 1000. Additionally, Swarm does not break batches on version boundaries. Delimiter case is currently not supported for version listing.
- *Simplified ACL management:*
When using per-object ACLs with versioning, the ACL for the current version of the object applies for determining authorization. To change the ACL for an object's entire version chain, update the object *without* specifying a version.

Supported Amazon S3 Features

This table summarizes the Amazon S3 features that are supported by the Gateway's S3 protocol implementation.



Note

When you are listing uploads and there are multiple simultaneous uploads in progress for a single object, only one of the uploads is in the listing.

Scope	Supported Operation
Error Responses	<ul style="list-style-type: none"> • HTTP Response Errors
Common Request Headers	<ul style="list-style-type: none"> • Authorization (AWS Signature Versions 2 and 4) • Content-Length • Content-MD5 • Content-Type • Date • Expect • Host • x-amz-date
Common Response Headers	<ul style="list-style-type: none"> • Connection • Content-Length • Content-Type • Date • ETag • Server • x-amz-delete-marker • x-amz-request-id
Service	<ul style="list-style-type: none"> • GET Service

Buckets	<ul style="list-style-type: none"> • DELETE Bucket • DELETE Bucket cors • DELETE Bucket policy • GET Bucket (list objects, v1 and v2) • GET Bucket acl • GET Bucket cors • GET Bucket Location • GET Bucket Object versions • GET Bucket policy • GET Bucket versioning • HEAD Bucket • List Multipart Uploads • PUT Bucket • PUT Bucket acl • PUT Bucket cors • PUT Bucket policy • PUT Bucket versioning • Cross-Region Replication (via Swarm replication feed)
Objects	<ul style="list-style-type: none"> • DELETE Object • DELETE Multiple Objects • GET Object • GET Object acl • HEAD Object • PUT Object • PUT Object acl • PUT Object - Copy • Initiate Multipart Upload • Upload Part • Upload Part - Copy • Complete Multipart Upload • Abort Multipart Upload • List Parts • Query String Request Authentication (pre-signed URLs)



Bucket PUTs

To support processes that require repeated bucket PUT requests to succeed, those requests return 409 Conflict, *regardless of owner*. This differs from AWS S3 behavior, which returns 403 Forbidden for non-owners.

S3 Application Integration

Configuring existing Amazon S3 applications to work with Swarm consists of changing the region end-point and changing the authentication credentials.



Best practice

Start with the [documentation provided by Amazon Web Services](#) and then use this section to help you integrate your S3 applications with the Swarm platform.

Within your S3 applications, change the following items:

1. **Region end-point** – Use the Swarm storage domain name in place of the Amazon S3 region end-point host name.
2. **Access Key** – In the Content UI, create an S3 authentication token in the correct domain and bucket, and use that token ID as your Access Key ID. For creating tokens in Content UI, see [Setting Tokens](#).
3. **Secret Key** – From the same token, use its secret key value in your S3 applications.
4. Update your configuration for best results:
 - Enable "path style" access to avoid certificate validation failure; otherwise, the client/SDK may attempt to access as `mybucket.mydomain.example.com/object`.
 - Use Version 2 signatures with the AWS .NET SDK for best performance. See <https://docs.aws.amazon.com/general/latest/gr/signature-version-2.html>.
 - Increase the part size to 100 MB or more for multipart uploads, if configurable.

S3 Protocol Configuration

Configure the Gateway as described in [Gateway Configuration](#) and then perform these additional steps to use the S3 front-end protocol:

1. Verify the [Swarm storage configuration settings](#) are correct, which is required for S3 clients to perform actions such as bucket deletion.
2. Edit the `gateway.cfg` file for S3 use:
 - a. Enable the S3 front-end protocol in the `[s3]` section.
 - b. Define `indexerHosts` for at least one indexer server in the `[storage_cluster]` section.
3. Create one or more [authentication tokens](#) for each S3 client.

When the S3 front-end protocol is in use, the Gateway must be able to query the Swarm Elasticsearch metadata index servers directly. Include as many as metadata index servers as needed in the `indexerHosts` parameter to spread the load and to provide fail-over in case one becomes unavailable.

The S3 protocol makes use of a shared secret key that is known to the client and the Gateway to provide request validation. The client creates an HMAC signature for every authenticated request and the Gateway must independently recreate the signature to validate the request. The AWS S3 access key and secret key is implemented with Gateway's [token-based authentication](#).

Storage SCSP Development

This section describes how to develop applications using the Swarm Simple Content Storage Protocol (SCSP), the mechanism that applications use to communicate with Swarm. SCSP is a text-based protocol based on the HyperText Transfer Protocol (HTTP) 1.1 standard. Using this section it is possible to:

- Create an application that connects to a storage cluster.
- Implement advanced HTTP features for improved performance.
- Map SCSP methods to HTTP methods.
- Implement erasure coding in large or unknown length objects.
- Implement SCSP methods, such as `READ`, `WRITE`, and `DELETE`.
- Manually create or rename a domain when the cluster administrator is not available.

- [Development Guidance](#)
- [Multipart Write](#)
- [SCSP Essentials](#)
- [Search Queries](#)
- [Connecting to a Swarm Cluster](#)
- [SCSP Methods](#)
- [Metadata Headers](#)
- [Content Integrity](#)

Development Guidance

This section highlights best practices and common pitfalls for developing application integrations with Swarm. Review these sections, and develop an understanding of Swarm objects and processes.

See [Swarm Concepts](#).

- [Common Integration Problems](#)
- [Application Best Practices](#)

Common Integration Problems

- [Stale Cache in a Large System](#)
 - [Important](#)
- [Workarounds for Redirect Issues](#)

Stale Cache in a Large System

Any client request can be directed to any node in the cluster because Swarm is a distributed system. Propagating actions takes time in a large distributed system. Certain types of transient errors can result after executing an SCSP "change" method, such as COPY, APPEND, UPDATE, and DELETE. This also applies to WRITE, which is not considered a change method.

An INFO request after performing an UPDATE on a bucket's metadata may return the old metadata. The error stops after about twice the value of the `cache.realmStaleTimeout` configuration parameter. `cache.realmStaleTimeout` is set to 10 minutes by default. After about twice that time – 20 minutes – following the APPEND, the bucket is visible by all nodes in the cluster.

Tasks that include these transient errors include:

- Using [SCSP UPDATE](#) to replace a bucket's metadata.
- Deleting a bucket and creating a new bucket with the same name.
- Using [SCSP COPY](#) to add custom metadata to a bucket or domain.

After about twice the value of the `cache.realmStaleTimeout` [configuration parameter](#) occurs, the same task should succeed.



Important

These examples apply to buckets and domains and not to named and unnamed objects. Before deleting a bucket, delete or move the objects it contains.

Workarounds for Redirect Issues

Some HTTP client libraries and frameworks do not handle redirects correctly for WRITE requests, at least not by default. Older versions of the HTTP protocol (namely HTTP 1.0) are lax in the specification of exactly what the client must do when it receives a 301 or a 307 response code. Because of this, older clients, including many web browsers, developed separate own conventions. While some of these conventions may be useful, they are in direct violation of the HTTP/1.1 specification and therefore incompatible with Swarm.

Both the Microsoft .NET framework and the libCURL framework (and probably others) are known to process a redirect from a POST request by changing the POST to a GET and then sending the new request to the redirect server. Workarounds exist in the impacted frameworks because the behavior is known to be in violation of the specification.

According to [RFC 2616](#): "When automatically redirecting a POST request after receiving a 301 status code, some existing HTTP 1.0 user agents will erroneously change it into a GET request."

Application Best Practices

- [Use an HTTP or S3 Library](#)
- [Protect Data in Transit](#)
- [Use Multithreading](#)
- [Maintain One Open Connection](#)
 - [Caution](#)

The following are important concepts and approaches for building optimal integrations to Swarm.

Use an HTTP or S3 Library

The Content Gateway is used to handle redirects internally and allows using it with a modern HTTP library or S3 SDK using SCSP or S3 API rather than connecting an application directly to Swarm nodes. Earlier [Swarm SDKs](#) (deprecated) are used to discover and connect to Swarm nodes for handling internal redirects, but Gateway provides additional functionalities such as authentication, authorization, and Swarm node connection pooling.

For example – It is suggested to use the popular *'requests'* library with Python. Use the official AWS SDKs on Java, C#, or Go programming language if an application uses the S3 API. Or to make SCSP requests, use a standard HTTP library like Apache HttpClient on Java.

Protect Data in Transit

The [Content-MD5 metadata header](#) provides an end-to-end message content integrity check (excluding metadata) of an object as it is sent and returned from Swarm.

A client application can:

- **Check this header** to detect modification of the object's body in transit.
- **Provide this header** to have Swarm compute and check it when storing or returning the data.

Swarm computes an MD5 digest during data transfer and then compares the computed digest to the one provided in the header if a Content-MD5 header is present on POST. Swarm returns a 400 Bad Request error response, abandons the object, and closes the client connection if the hashes do not match.

Content-MD5 headers are stored with the object metadata and returned on all subsequent GET or HEAD requests. Swarm computes the hash as the bytes are read if a Content-MD5 header is included with a GET request. The connection is closed before the last bytes are transmitted, which is the standard method to indicate something went wrong with the transfer if the computed and provided hashes do not match.

The Content-MD5 header provides an extra level of insurance, protecting against potential damage in transit as well as from damage while in storage.

See [Configuring Swarm Storage](#) for [configuration parameters](#) and *how to edit the configuration files*.

See [Content-MD5 Checksums](#).

See [Lifepoint Metadata Headers](#) for *more information about lifecycle management*.

See [Content Integrity Assurance](#).

Use Multithreading

Swarm is a multithreaded, multi-node cluster. Every node in a storage cluster can establish and maintain connections with many different client applications at the same time. Normally an application opens one SCSP connection to the cluster and sends requests and receive responses in a sequential manner. A single client application may choose to open more than one connection to Swarm to achieve better response times and read or write throughput for high-volume applications.

This multithreaded client strategy can be very effective in improving overall performance when necessary because Swarm automatically load balances requests by causing them to be redirected to a less busy node in the cluster that is capable of servicing the request. Each client thread (or process) can be connected to different nodes within the cluster.

Maintain One Open Connection

The Swarm software implements [HTTP/1.1 persistent connections](#). That means a client application is not required to close the socket or connection after each request. Swarm holds connections open and allows the client to continue sending requests and receiving responses until either the client closes the connection explicitly, or it stops sending requests for some period of time.

The client *must* close the connection and reopen a new connection when a Swarm response includes the header **Connection: close**. This is performed when there is an error that causes confusion as to the meaning of the remaining bytes sent over the connection.

Have the client maintain one open connection at a time using one of these methods:

- Close the old connection before opening the new, redirected connection.
- Maintain a pool of connections to several nodes in the cluster. The pool approach can considerably improve response times because the client eventually has open connections to all nodes for smaller clusters.



Caution

Do not to exceed the operating system limits on the number of simultaneously open connections for very large storage clusters.

Multipart Write

Note

Multipart Write was previously referred to as *Parallel Write*; the functionality is the same.

Parts of a large object from multiple clients at the same time can be uploaded with Multipart Write. Multipart Write allow client applications to split a large file in to multiple pieces, transfer the pieces concurrently to Swarm, and then request Swarm combine the separately uploaded parts together as a single object, thereby minimizing the upload time.

Multipart write requires [erasure coding](#) (EC). The health processor (HP) has the ability to consolidate segments of erasure-coded objects with sub-optimal segment usage, as can happen when performing SCSP or S3 multipart writes of objects using small parts. DataCore recommends configuring clients to use 50MB-100MB parts. Set the configuration setting, [ec.segmentConsolidationFrequency](#), to 10 (recommended), which performs all consolidations over 20 HP cycles if consolidation is needed.

Tip

Every multipart write must be erasure-coded for upload; the HP converts it to a replicated object if the uploaded object does not meet the current policy for EC encoding. Add a [lifepoint](#) to that effect to maintain erasure coding for the lifetime of the object.

Three distinct actions must be performed in this order to upload a large object in parts using multipart write:

1. Initiate a multipart write.
 2. Upload or copy the parts.
 3. Complete or cancel the procedure.
- [Uploading the Parts](#)
 - [Initiating Multipart Write](#)
 - [Canceling a Multipart Write](#)
 - [Completing the Multipart Write](#)
 - [Multipart Write Example](#)
 - [Validating a Multipart Write](#)

Uploading the Parts

- [Using object part numbers](#)
- [Uploading a part](#)
- [Uploading a part by copying from an existing object](#)
- [Validating the uploaded parts](#)

The individual parts of a larger object are ready to upload when the initiate request is complete. Swarm allows any number of parts in a multipart upload. (v9.1)

Create a POST request with the object name used in the initiate request, the upload ID returned from the initiate request, and a unique part number for each part to upload a part.

The part uploads for an upload ID must include the same domain query argument if the initiate request included a domain query argument for a specific domain. The part uploads must not include an encoding query argument as they inherit whatever was specified in the initiate request unlike the initiate request. A failed part upload can be retried without affecting the outcome of the multipart upload.

A part is stored as an immutable object whose **Content-UUID** is returned in the request response when it is successfully uploaded. Client applications must keep track of the part number used for the upload and the **Content-UUID** Swarm assigned when stored to eventually complete the multipart upload, as described below.



Parts are unnamed

Each *part* is an immutable object that returns a Content-UUID even if the initiated object is named, even though a POST on a named object does not ordinarily return that header. The parts are tenanted in the same domain as the destination object, but parts are unnamed, so they cannot reside in buckets.

Using object part numbers

Swarm uses part numbers to identify the position of each part in an object. Include the upload ID and a unique part number for each part so Swarm can assemble the parts in the correct order when uploading parts. Non-sequential part numbers for each part can be selected (2, 4, 6, 8), but Swarm assembles the parts in sequential order.

Record each part number and corresponding **Content-UUID**. This information is required to [complete the multipart write procedure](#).

Uploading a part

The following must be included to upload each part:

1. The object name or UUID (or Content-UUID, for [immutable objects](#)) returned by the initiate request
2. The upload ID returned from the initiate request
3. A unique part number for each part uploaded

```
POST /ObjectNameorUUID?partNumber=PartNumber&uploadId=UploadId HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
Content-Length: 43402
Expect: 100-continue
[ content ]
```

The content to be uploaded for the part is placed in the body of the request, like a normal POST operation.

Uploading a part by copying from an existing object

Create a POST request that uses the content from existing objects if the required parts currently exist in the storage cluster. Swarm creates an EC copy of the object for the multipart write when the part copy request is completed. The request fails if the source object does not exist or cannot be read from the specified range. This process leaves the source and destination versions unrelated to each other.



Content Gateway

The user making the "PUT with copy" request must have read access to the source object when going through Gateway.

The additional headers used in this request specify the source object and the range in bytes. The source object must be specified in the **x-castor-copy-source** header by UUID or object name in bucket/object-name format:

```
curl -i "$HOST/fd9cf39f056fb0dd858d8fb288c22885?PartNumber=3
  &UploadID=ddd080eb400bd5531f580191e3c5a916dd66c7c1e3244dc6cad46183097677e6dd66c7c1e3244dc6cad46
-XPOST
-H "x-castor-copy-source: a08212d59b5bd306a52008dfef335be2"
-H "x-castor-copy-source-range: 5-8"
```

```
HTTP/1.1 201 Created
Location: http://192.168.1.171:80/09938e338c3590b93855d7cca2179aec
Location: http://192.168.1.109:80/09938e338c3590b93855d7cca2179aec
Volume: 3f5ef63dab992ebcf28e092bb56103c3
Volume: 12e08e29145f277501a6490b602ea287
Manifest: ec
Castor-System-UploadID: ddd080eb400bd5531f580191e3c5a916dd66c7c1e3244dc6cad46183097677e6dd66c7c1e
Content-UUID: 09938e338c3590b93855d7cca2179aec
Last-Modified: Tue, 27 Sep 2016 20:49:46 GMT
Entity-MD5: 9g0GoVLSYSXc/PMI4FWKbQ==
Stored-Digest: f60d06a152d26125dcfcf308e0558a6d
Castor-System-Encoding: zfec 1.4(1, 1, 524288, 200000000)
Castor-System-Version: 1475009386.549
Etag: "4c760a34ee534bcdba91680919378e2e"
Content-Range: bytes 5-8/10
Replica-Count: 2
Date: Tue, 27 Sep 2016 20:49:46 GMT
Server: CAStor Cluster/8.2.a
Content-Length: 46
Content-Type: text/html
Keep-Alive: timeout=14400

<html><body>New stream created</body></html>
```

The **Content-MD5** is applied to the range read if a **gencontentmd5** query argument (or the deprecated Expect: Content-MD5 header) is applied to a part copy with a range read.

These are the arguments and headers that are required for a part upload request that copies data from an existing object:

```
POST /ObjectNameorUUID?partNumber=PartNumber&uploadId=UploadId HTTP/1.1
x-castor-copy-source: uuid/name
x-castor-copy-source-domain: domain_name
```



Response Headers

The result code for the operation (which always responds with chunked encoding) is in the trailing header **Castor-System-Result** when the **x-castor-copy-source** header is used.

See "Response Headers for Multipart Writes" in [Completing the Multipart Write](#).

-copy-source- headers

The following headers in the POST request for part uploads are optional, except for the first. They perform the same filtering as the [regular headers of those names](#) (**range**, **if-match**, etc.) performed against the source object being copied if included.

ⓘ **Note**

Error responses on conditional headers come back immediately, in place of a 202 (Accepted for processing) response. Condition failures (such as the ETags not matching) are reported in the initial HTTP response, not the castor-system-result header.

Type	Header	Notes
Source	x-castor-copy-source	Required. Must be a valid name or UUID.
Domain	x-castor-copy-source-domain	Required (unless untenanted)
Range	x-castor-copy-source-range: bytes=first-last	If the range values are out of bounds for the data, the request returns 416 (Range Not Satisfiable). <i>Tip:</i> To copy from a start range to the end of the object, omit the end range. (v11.1)
Conditional	x-castor-copy-source-if-match: "<ETag>" x-castor-copy-source-if-none-match: "<ETag>"	The ETag <i>must</i> be enclosed in quotes.
	x-castor-copy-source-if-unmodified-since: <timestamp> x-castor-copy-source-if-modified-since: <timestamp>	Uses the format of the standard HTTP last-modified header.

Validating the uploaded parts

Include a **gencontentmd5** query argument or **Content-MD5** header in the POST argument to return a **Content-MD5** header to validate the content of the uploaded part. See [Content-MD5 Checksums](#). (v9.2)

Verify each part has an MD5 stored with it if intending to validate the full transfer on the complete using the Composite-Content-MD5 header.

Initiating Multipart Write

- [Important](#)
- [Multipart writing a new named object](#)
- [Multipart writing an immutable object](#)
- [Multipart writing an existing object](#)

A multipart, parallel upload is started with an initiate request that can be an SCSP POST, PUT, or APPEND request. If you are appending to an existing object, the object must already be erasure-coded or the request fails. To convert a replicated object to EC, use a 0-byte APPEND with the query arguments `erasurecoded=yes` and `encoding=<k:p>`.

Query argument - The initiate request must include the `uploads` query argument, and it can specify an immutable, named, or aliased object. The object encoding is determined by the `encoding` query argument; if the argument is missing, the applicable [encoding policy](#) stands. Any body text included on the initiate request is ignored and not included in the final object.

Upload ID - When the request is completed, Swarm returns an upload ID that identifies the upload. This ID serves as the identifier for all subsequent operations associated with the upload. The name or resulting Content-UUID value from the initiate request is the object name used for subsequent operations for the same upload ID.



Important

Do not include a Content-MD5 header in the initiate or complete request. This operation is unsupported and returns a 400 Bad Request error response. To generate a Content-MD5 for an individual part or the completed object, use a separate POST request with the `gencontentmd5` [query argument](#).

Multipart writing a new named object

Example of initiating a multipart write for a named object:

```
POST /exampleBucket/objectName?domain=yourDomainName&uploads&encoding=5:2 HTTP/1.1
x-custom-header1-meta: value
x-custom-headerN-meta: value
```

The custom header arguments are included because any custom metadata for the object must be declared with the initiate request. Any content in the body of the request is ignored when the multipart upload is eventually completed.

When complete, Swarm responds with a header that provides an upload ID. Record this ID so you can upload the object parts in subsequent requests.

Example of a header with an upload ID:

```
Castor-System-UploadId: VXBsb2FkIEEIEIG...5tMnRzIHVwbG9hZA
```

Multipart writing an immutable object

Example of initiating a multipart write for an unnamed, immutable object:

```
POST /?uploads&encoding=5:2 HTTP/1.1
x-custom-header1-meta: value
x-custom-headerN-meta: value
```

The custom header arguments are included because any custom metadata for the object must be declared with the initiate request. Any content in the body of the request is ignored when the multipart upload is eventually completed.

When complete, Swarm responds with a header that provides an upload ID. Record this ID so you can [upload the object parts](#) in subsequent requests.

Example of a header with an upload ID:

```
Castor-System-UploadId: VXBsb2FkIEEIEIG...5tMnRzIHVwbG9hZA
```

Multipart writing an existing object

In addition to creating a new object in the cluster with multipart write, you may overwrite an existing aliased or named object with multipart upload via a `PUT` request or add data to an existing object via an `APPEND` request.

The existing object remains distinct and independent until the multipart write operation is completed. When completed, the existing object is replaced by the new object. If you complete the operation with an `ABORT` request, the existing object remains unchanged.

Canceling a Multipart Write

A DELETE request can be used with the [UploadID query argument](#) to cancel an in-progress multipart write. This method enables the Health Processor to delete any completed or in-progress parts. Canceling a multipart write does not terminate any part uploads still in progress.

Updates – The deletion deletes the multipart write and leaves the original object intact if canceling a multipart write intended to modify an existing object.



Warning

Swarm attempts to delete the *original* object if the `UploadID` is no included on the abort deletion for an object previously existing in the cluster. The `uploadId` query argument must be included to verify the multipart write operation is deleted.

Example

Example of canceling a multipart write:

```
DELETE /ObjectName?uploadId=UPLOAD_ID HTTP/1.1
```

- **Success:** Swarm returns a 200 OK code when the cancel is completed.
- **Failure:** Any failure to write one of the parts triggers an immediate upload cancellation. The upload is cleaned up and retried as a whole at a later point in time.

Completing the Multipart Write

- [Changes during the write](#)
- [Requesting Completion](#)
 - [Parallel upload complete request for 3-part object](#)
 - [Important](#)
- [Ordering the Parts](#)
 - [Parallel upload complete request with manifest order](#)
- [Response Headers for Multipart Writes](#)
 - [Initiation Response](#)
 - [Completion Response](#)

The multipart write procedure is completed using a POST request with the `?uploadId=<uploadid>` query argument and no `?partNumber`. Include a list of all individual parts in JSON format with the Content-UUID and part numbers recorded for each object when the parts are uploaded in the body of the request. The completion request fails if multiple parts with the same part number are accidentally included.

Swarm creates the new object or modifies the existing object by assembling the uploaded parts in ascending part-number order (the default) or by the manifest order when the multipart write is completed. An existing object is modified as specified in the initiate request, either overwriting all data with a PUT or adding the multipart upload data with an APPEND.



Changes during the write

The following operations occur when an existing object is the target of a multipart write and changes to that object occur *before* the write completes:

- PUT – the completion operation fails.
- APPEND – the operation appends to the content as it exists upon completion, not as it existed when it was started.

Requesting Completion

The completion request must not include an encoding query argument, as it inherits whatever was specified on the initiate request. Custom metadata may be provided with the complete request and is merged with the metadata provided in the initiate request. The metadata from the initiate request takes precedence if the metadata in the two requests collide.

Parallel upload complete request for 3-part object

POST /exampleBucket/ObjectName?uploadId=UploadId HTTP/1.1

```
{
  "parts": [
    {
      "partNumber": 1,
      "uuid": "12345678901234567890123456789012"
    },
    {
      "partNumber": 2,
      "uuid": "12345678901234567890123456789013"
    },
    {
      "partNumber": 3,
      "uuid": "12345678901234567890123456789014"
    }
  ]
}
```

Important

Verify the part numbers and corresponding Content-UUIDs of all required parts are included in the manifest before the request is completed. Any part not included with the manifest is not used in the final object and is eventually be deleted by the Health Processor.

Ordering the Parts

Parts are ordered by part number by default. Use the global value **sortOrder** set to **natural** (which is the manifest order) if the parts need to be reassembled according to the order they appear in the manifest rather than by the part number.

Parallel upload complete request with manifest order

POST /exampleBucket/ObjectName?uploadId=UploadId HTTP/1.1

```
{
  "sortOrder" : "natural",
  "parts": [
    {
      "partNumber": 2,
      "uuid": "12345678901234567890123456789013"
    },
    {
      "partNumber": 3,
      "uuid": "12345678901234567890123456789014"
    },
    {
      "partNumber": 1,
      "uuid": "12345678901234567890123456789012"
    }
  ]
}
```

Response Headers for Multipart Writes

Swarm uses *chunked transfer encoding* to verify the client socket remains open throughout a lengthy complete for multipart write operations. Chunked transfer encoding is the streaming data transfer mechanism in HTTP/1.1. The chunking method allows content to be transferred iteratively along with the information needed to verify when it has been received in full. The method is specified by this response header:

```
Transfer-Encoding: chunked
```

The data is divided into non-overlapping "chunks" sent and received independently in chunked encoding. The encoding modifies the message body to transfer it as a series of chunks, each of which is preceded by a size indicator (in bytes). The transmission ends when a 0-length chunk is received, which can be followed by a *trailer* that's terminated by an empty line. Swarm uses the trailer to send additional HTTP header fields with information about the multipart operation completion.

Initiation Response

These are the typical results and relevant headers for the *initiation* of the multipart operation:

Castor-System-Result	Additional Headers	Notes
400 (Bad Request) 404 (Not Found) ...	Castor-System-Error-Code Castor-System-Error-Text Castor-System-Error-Token	The result indicates warm found an error before <i>starting</i> the completion process, such as a problem with the parts manifest. See Error Response Headers .
202 (Accepted)	Completion-Etag	The result indicates Swarm has accepted the uploaded parts and begins assembling them into a new object. The initial response includes a <code>Completion-Etag</code> with the value of the <i>expected</i> ETag for S3 compatibility. There is no new object and this ETag provided is not valid if there is an error later. (v11.1)

Completion Response

These are the typical results and relevant headers for the *completion* of the multipart operation:

Castor-System-Result	Additional Headers	Notes
400 (Bad Request) 404 (Not Found) ...	Castor-System-Error-Code Castor-System-Error-Text Castor-System-Error-Token	The result indicates Swarm experienced an error that prevented the process from completing. See Error Response Headers .
201 (Created)	Completion-Etag	The result indicates Swarm succeeded in creating a new object from all parts. The additional header specifies the ETag of the new Swarm object constructed from the parts. The result includes Location headers of the resulting manifests that are analogous to the Location headers of a normal EC write.

Multipart Write Example

Following is an example of the entire multipart object uploading POST process, using CURL (with line breaks for ease of reading).

i CLUSTER or <cluster> in a URL stands for <host>[:<port>], where *host* is a fully qualified domain name or IP address, plus a *port* number if other than 80. If the Host header does not match the domain name, override it with the *domain=* argument.

1. Start the upload with a 0-byte POST. Set the encoding (&encoding=2:1) if there is no default encoding or you need a non-default setting:

Initiate the upload

```
curl -i
  "${cluster}/
  ?uploads"
-XPOST
```

2. In the response, locate the headers for the **UploadID** and the **Content-UUID**, which is the new object being created:

Returned headers

```
Castor-System-UploadID:
 c88fe8a5daf98f7ce84dc4947238f5c1d0b0b42b8ce464e4566cc2c080ecf401d0b0b42b8ce464e4566cc2c080e
Content-UUID:
 c88fe8a5daf98f7ce84dc4947238f5c1
```

3. Write the first part using the **UploadID** and **Content-UUID** and **partNumber=1**:

Upload the first part

```
curl -i
  "${cluster}/c88fe8a5daf98f7ce84dc4947238f5c1
  ?partNumber=1
  &uploadid=c88fe8a5daf98f7ce84dc4947238f5c1d0b0b42b8ce464e4566cc2c080ecf401d0b0b42b8ce464e4!
  -XPOST
  --data-binary
  "part 1 data"
```

Use the Content-UUID returned by this post in the manifest to complete the upload.

4. Start the second part using the **UploadID** and **Content-UUID** and **partNumber=2**:

Upload the second part

```
curl -i
  "${cluster}/c88fe8a5daf98f7ce84dc4947238f5c1
  ?partNumber=2
  &uploadid=c88fe8a5daf98f7ce84dc4947238f5c1d0b0b42b8ce464e4566cc2c080ecf401d0b0b42b8ce464e4!
  -XPOST
  --data-binary
  "part 2 data"
```

Use the Content-UUID returned by this post in the manifest to complete the upload.

5. Complete the upload with the **UploadID** and a part manifest:

i **Note**
Follow this usage of double quotes.

Complete the upload

```
curl -i
"${cluster}/c88fe8a5daf98f7ce84dc4947238f5c1
?uploadid=c88fe8a5daf98f7ce84dc4947238f5c1d0b0b42b8ce464e4566cc2c080ecf401d0b0b42b8ce464e4!
-XPOST
--data-binary '{
  "parts":[
    {
      "partNumber":1,
      "uuid":"9b149f0959839cee2c915dedfa8d7e25"
    },
    {
      "partNumber":2,
      "uuid":"76e7f0c7c417b7bca7daedbe3e18bf40"
    }
  ]
}'
```

- The response code on the complete indicates success. To verify the completed object, you can HEAD the object:

Verify completed object

```
curl --head "${cluster}/c88fe8a5daf98f7ce84dc4947238f5c1"
```

Validating a Multipart Write

- [Validation with Composite-Content-MD5](#)
 - [Calculating a Composite-Content-MD5](#)
 - [Composite-Content-MD5 Example](#)
- [Validation with Content-MD5](#)

Validation with Composite-Content-MD5

Swarm enables transfer validation on multipart requests by way of a composite MD5 computed from the Content-MD5 hashes from all parts. The request header that enables multipart validation is `Composite-Content-MD5`. This header provides an end-to-end integrity check of the content (excluding metadata) of a multipart write request at the time of completion. The value can be used to validate the object contents if all parts are stored with a Content-MD5.

- **Storage** - The value of the composite MD5 is persisted and indexed in a header called `Castor-System-CompositeMD5` in the metadata section of the completed object's manifest. It is not preserved across a PUT or APPEND, but it is automatically persisted across a COPY so MD5 need not be recalculated on very large files, which is inefficient.
- **Behavior** - On a complete request that includes the `Composite-Content-MD5` header, Swarm computes the value for the overall request from the MD5 of the concatenated Content-MD5 values stored with each part (in order).
 - If a `Composite-Content-MD5` header is sent in the request, Swarm must calculate and compare it with the stored value. The complete request can succeed only if the composite value Swarm calculates matches the value provided on the header.
 - On the completion response, the `Castor-System-CompositeMD5` header is provided as a trailing header if the Content-MD5 is available on all parts, regardless of whether `Composite-Content-MD5` is provided on the request.
 - For newly completed multipart writes, the `Composite-Content-MD5` header is also indexed in Elasticsearch, so it appears in listings:

```
curl 'https://www.example.com/mybucket?format=json\
&fields=name,tborn,etag,content-md5,Castor-System-CompositeMD5'
[ {
  "last_modified": "2017-04-08T20:37:02.868400Z",
  "castor_system_compositemd5": "306cca04302861ed2620a328f286346f-5",
  "hash": "ae478cc4c3eb28b432825074673aeda9",
  "name": "samples/5G"
}]
```

(v9.2)

- **Usage** - Pass in a composite MD5 made by taking the md5 of the concatenation of the binary md5 of the parts, in order, with no gaps, providing it as the value of the `'Composite-Content-MD5'` header. This triggers Swarm to collect the Content-MD5s from each part and to assemble its comparison value.
- **Failure** - If the calculated value does not match the supplied value, or if any part is missing a Content-MD5 header, the request fails with a 409 (Conflict). In this case, review the error message and correct the problem (such as parts missing the Content-MD5 header) before attempting the complete again.

Calculating a Composite-Content-MD5

These are different ways to represent an MD5 hash:

Base64	rbyRpD6YijtbdFuFKakLYQ==	Binary-to-text encoding
---------------	--------------------------	-------------------------

hexdigest	adbc91a43e988a3b5b745b8529a90b61	HEX string representing the hash base64.b64decode('rbyRpD6YijtbdFuFKakLYQ==').encode('hex')
------------------	----------------------------------	--

For Composite-Content-MD5, you need to end up with the *HEX digest* of the MD5 hash of the concatenated binary MD5 hashes of all parts, in order. The composite value starts with the hex digest of that hashed concatenation of hashes, followed by a hyphen and the number of parts:

{ hash of concatenated part MD5 hashes, in order }-{ number of parts }	754e6c52092a9c1134d7f047d61db168-3
{ hash of part1hash & part2hash & ... & partnhash }-{ n }	

A multipart object with three parts:

- Part 1 Content-MD5 = rbyRpD6YijtbdFuFKakLYQ==
- Part 2 Content-MD5 = 9lzbDNFcX99eTYqZB4QKjg==
- Part 3 Content-MD5 = 2qHK6cuQufMzJAs6lxTmKQ==

The composite hash is this:

- Composite-Content-MD5 = 754e6c52092a9c1134d7f047d61db168-3

Calculating it involves this type of process:

```
partBinaryMD5-1 = base64.b64decode( contentMD5HeaderValue1 )
partBinaryMD5-2 = base64.b64decode( contentMD5HeaderValue2 )
partBinaryMD5-3 = base64.b64decode( contentMD5HeaderValue3 )
Composite-Content-MD5 = hashlib.md5("".join([ partBinaryMD5-1, partBinaryMD5-2, partBinaryMD5-3 ]
```

Composite-Content-MD5 Example

1. Given a file divided into three parts, get the hex md5 digest for each part using `md5sum`.

- Part 1 hash: babfc3ceb8a4568587b7d31bfff36257
- Part 2 hash: fae6c82883c12e289bc5f12f3ecf76ef2
- Part 3 hash: 2afdd827a9e785029f9692e82ea07cca

2. Concatenate the MD5s together into a new file.

```
echo "babfc3ceb8a4568587b7d31bfff36257" >> md5sums.txt
```

3. See the result by catting the file:

```
cat md5sums.txt
babfc3ceb8a4568587b7d31bfff36257fae6c82883c12e289bc5f12f3ecf76ef2afdd827a9e785029f9692e82ea07cca
```

4. Convert to binary and hash it to get the composite MD5 using `xxd` and `md5sum`.

```
xxd -r -p checksums.txt | md5sum
12138b95c0af8f8e764f80d719cc7cbd -
```

5. Append the part count (3) to get the final composite header value:

```
12138b95c0af8f8e764f80d719cc7cbd-3
```

6. Use the value to complete the multipart write:

```
curl -v "192.168.3.84/8c249211d4dc9683ab005760675b7c46?
uploadId=8c249211d4dc9683ab005760675b7c460372fa38bbaad17fefc9883c48ed55b80372fa38bbaad17fefc9883
-L --post301
-H "Composite-content-MD5:12138b95c0af8f8e764f80d719cc7cbd-3"
-d '{ "parts" : [
  { "partNumber":1, "uuid":"66d6b90b7f451f2a5ad644884d1f9106"}
  /
  { "partNumber":2, "uuid":"5eefe725084eb6ba02cf146400f50ec4"}
  /
  { "partNumber":3, "uuid":"64cdble3457a69c75ad932ad9661e93d"}
]}'

POST /8c249211d4dc9683ab005760675b7c46?
uploadId=8c249211d4dc9683ab005760675b7c460372fa38bbaad17fefc9883c48ed55b80372fa38bbaad17fefc9883
HTTP/1.1
User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.21 Basic ECC zlib/1.2.3 li
libssh2/1.4.2
Host: 192.168.3.84
Accept: /
Composite-content-MD5:12138b95c0af8f8e764f80d719cc7cbd-3
Content-Length: 198
Content-Type: application/x-www-form-urlencoded

HTTP/1.1 202 Accepted
...
castor-system-result: 201
```

Validation with Content-MD5

Use a GET request with the `gencontentmd5` query argument and compare the result with the known value to validate an object created by a completed multipart write. See [Content-MD5 Checksums](#). (v9.2)

SCSP Essentials

- [SCSP as a subset of HTTP](#)
- [Mapping SCSP to HTTP methods](#)
- [SCSP protocol](#)

This overview of the Simple Content Storage Protocol (SCSP) methods explains how they map to the corresponding HTTP methods.

SCSP as a subset of HTTP

The mechanism that applications use to communicate with Swarm is a text-based protocol based on HTTP. Known as the **Simple Content Storage Protocol (SCSP)**, the methods and syntax are a proper subset of the HTTP/1.1 standard.

Although many of the optional parts of HTTP/1.1 are not implemented in SCSP (which is why the protocol is referred to as *simple*), all required protocol components are implemented, as well as several of the common methods.

Swarm assumes communication with an HTTP/1.1 compliant client application.

See [Working with Large Objects](#).

See the [SDK Overview](#) for the API-level implementation of SCSP. The SDK helps developers write integrations to Swarm. The SDK includes sample code in Java, Python, C++, and C#.

Mapping SCSP to HTTP methods

The following table maps SCSP methods to complementary HTTP methods.

SCSP Method	HTTP Method	RFC 7231 Section
READ	GET	4.3.1
INFO	HEAD	4.3.2
WRITE	POST	4.3.3
UPDATE	PUT	4.3.4
DELETE	DELETE	4.3.5
n/a	CONNECT	4.3.6
n/a	OPTIONS	4.3.7
n/a	TRACE	4.3.8
APPEND		
COPY		

SCSP protocol

Most HTTP communication is initiated by a client application and consists of a request to be applied to an object on a Swarm server. This is performed using a single connection between the client application and the Swarm server. Being HTTP-based, SCSP protocol consists of HTTP requests and responses:

- **Requests** are generated by a Swarm client (that is, any HTTP/1.1 client), with these components:
 - Request method, with URI and protocol version
 - Case-insensitive [query arguments](#)
 - Required and optional [headers](#)
- **Responses** are generated by one or more nodes in a storage cluster, with these components:
 - Status line, with the message's protocol version and a success or error code
 - MIME-like message, with server information, entity metadata, and possible entity-body content

See the [HTTP/1.1 specification](#) for the semantics and nuances of HTTP.

- [SCSP Headers](#)
- [Undefined Responses from Swarm](#)
- [Formatting SCSP Commands](#)
- [SCSP Compatibility and Support](#)
- [Error Response Headers](#)
- [SCSP Query Arguments](#)
- [HTTP Response Codes](#)

SCSP Headers

- [Best practice](#)
- [Swarm-Specific Headers](#)
- [Standard HTTP Headers](#)

Header limits – Headers (metadata) are constrained not only by Swarm but by all services, proxies, and clients (such as Elasticsearch, Twisted, Jetty, HAProxy) handling objects. A Swarm object may have any number of [annotations](#) associated with it, but each annotation object is subject to these limits. The persisted metadata on Swarm objects must fall within these limits by default:

- 500 – The *total number* of headers on an object
- 32 KB – The *combined length* of all headers (key/value pairs); exceeding this returns a 400 (Bad Request) error
- 16 KB – The *maximum length* for a given header (key/value pair)

The following tables list all headers supported or used by Swarm. The headers described apply to buckets, named objects, and aliased objects unless otherwise noted.

- **Methods** are the SCSP methods to which the header applies.
- *Request* (signified by X)
- *Response* (signified by X)
 - **V** means it appears *only with* `verbose=true`
 - **T** means it appears in a trailing response
- *Writable* (signified by W)
 - **R** means it is read-only
- *Persisted* indicates whether the header is persisted with the object. Note: some persisted headers are not writable (signified by P).



Best practice

Always use **case-insensitive** header matching.

Swarm-Specific Headers

Header	Description	Method	Request	Response [Trailing] [Verbose]	Writable [Read- only]	Persisted
Castor-System-Accessed	Used on GET/HEAD requests to indicate the time the object was last accessed (written or GET).	GET, HEAD		X V		
Castor-System-CompositeMD5	Multipart-written objects only. This composite value, stored with the completed object, is computed on the multipart complete as the sum of the Content-MD5 values of all parts. This header value is reused on subsequent COPY operations but removed on other updates (PUT, APPEND).	GET, HEAD, POST		T		

Castor-System-Decorates	Set to the ETag of a Swarm object to be decorated. When the decorated object is deleted or overwritten, the object with this header is reclaimed by the health processor.	All	X	X	W	P
Castor-System-Headers-Filtered	If response headers are filtered by a whitelist or blacklist, this header is added.	GET, HEAD		X	R	
Castor-System-InProgress	Returned with value yes on a verbose request when the inprogress flag is set on a stream. Not returned at all when the flag is not set.	GET, HEAD		X V		
Castor-System-Result	Reports the response code, such as 201 (Created) or 404 (Not Found), as a trailing header for a multipart write completion.	POST		T		
Castor-System-Tiered	Audit headers for objects restored by an S3 Backup feed. One copy captures the date and source cluster of the backup, and another captures the date and S3 host/bucket of the restore. Dates are same format as Last-Modified.	GET, HEAD		X V	R	P
Castor-{anything}	{anything} cannot start with "system". Uninterpreted; meant for client customization.	All			W	P
Castor-Authorization	Content-level authorization.	All	X		W	P
Castor-System-Bytes-Used	With bucket list du, the number of bytes in the query without replicas.	GET (listing)		X		
Castor-System-Bytes-Used-With-Reps	With bucket list du / withreps requests, the number of bytes in the query with replicas.	GET (listing)		X		
Castor-System-Object-Count	In a bucket listing request, the number of objects returned by the query.			X		
Castor-System-Alias	The alias UUID of an alias object, bucket, or domain. This UUID cannot be used in any SCSP method on a domain, bucket, or named object. Use it to execute SCSP methods on an aliased object or to delete an unnamed immutable object.	GET, HEAD, DELETE		X	R	P
Castor-System-Auth	Special headers for a GET/retrieve request. The username:password for an administrative account on the remote cluster, only if it differs from the source cluster.	SEND	X			
Castor-System-CID	For named and tenanted unnamed objects, the alias UUID of the owning context.	GET, HEAD, DELETE		X	R	P
Castor-System-Cluster	Name of the cluster where the object was created or last updated. For SEND, The value of the cluster.name setting of the destination cluster.	GET, HEAD, DELETE, SEND		X	R	P
Castor-System-Created	A timestamp that specifies when the object was created or last updated. Uses HTTP time, an ASCII format based on GMT.	GET, HEAD, DELETE		X	R	P

Castor-System-Domain	Named objects only. Shows the name of the domain where the object was created. The domain name in which this object is tenanted. Computed on GET and HEAD. Returns <domain>.	GET, HEAD		X V	R	P
Castor-System-EnforceTenancy	True or False depending on cluster. enforceTenancy. A response header on the status page.	GET		X		
Castor-System-Error-Code	Sent with any error response. The request error code (if applicable). This code is usually a 4xx or 5xx HTTP response code. May appear as a trailing header on multipart write complete requests.	All		X		
Castor-System-Error-Text	Sent with any error response. The request error description (if applicable). Provides a description of the error. May appear as a trailing header on multipart write complete requests.	All		X		
Castor-System-Error-Token	Sent with any error response. A unique error token (if applicable). Provides a parsed token that uniquely identifies the error. May appear as a trailing header on multipart write complete requests.	All		X		
Castor-System-IsVersioned	Header indicating the object versioning state in effect at the time the object was written. May only appear on alias or named objects. True if the object was written in the versioning-Enabled state. "False" if the object was written in the versioning-Suspended state. There is no header otherwise. A flag for the object's Swarm versioning status, which does not appear on objects created in a versioning-disabled context. True means it was created in the versioning-Enabled state. False means it was created in the versioning-Suspended state. 9.6.a: Maps to "versioned" (boolean) search query argument.	GET, HEAD			R	P
Castor-System-LicenseCapacityTB	Return on a cluster status request. Indicates the licensed capacity, in TB.			X		
Castor-System-LicenseSerialNumber	Returned on a cluster status request. Indicates the license serial number, or 'unregistered.'			X		
Castor-System-Name	Named objects only. Symbolic name of the object, which is the user-Specified (partial) name for this object.	GET, HEAD, DELETE		X	R	P
Castor-System-Next-Version	Appears only on admin or administrative GET or HEAD responses. When object versioning is enabled or suspended, named and alias objects include these headers if there is a previous or next version in the version chain for the object.	GET, HEAD		X V		
Castor-System-Owner	The user id of the authenticated user who wrote the object. Name of the user who created or last modified the object. If the object was created anonymously (that is, without authenticating), this header is absent.	All			R	P

Castor-System-PartNumber	Special headers for multipart write requests. Identifies the part number of a part in the temporary manifest for a multipart write.	POST, PUT	X		W	P
Castor-System-Path	For named objects only, shows the full path of the object in /domain/bucket/object format. Example: /cluster.example.com/mybucket/mypath/myobject.html. Computed on GET and HEAD. Returns <domain>/<bucket>/<name>.	GET, HEAD		X V	R	P
Castor-System-Persisted-Headers	Indicates which of the other headers on the response are persisted.	GET, HEAD		X V		
Castor-System-Previous-Version	Appears only on admin or administrative GET or HEAD responses. When object versioning is enabled or suspended, named and alias objects include these headers if there is a previous or next version in the version chain for the object.	GET, HEAD		X V		
Castor-System-TotalGBAvailable	A response header on the status page.	GET		X		
Castor-System-TotalGBCapacity	A response header on the status page.	GET		X		
Castor-System-UploadID	Special header for multipart write requests. As a response header, it identifies the multipart write request for all subsequent requests relating to the upload. As a persisted header, it associates a stream with a multipart upload, either as the init stream or a part, based on its Castor-System-PartNumber value.	GET, HEAD	X		R	P
Castor-System-Version	A numerical version number associated with mutable objects. This is a UNIX-Style floating point time since GMT epoch with millisecond accuracy. The timestamp (in seconds since epoch) when the object was written, which corresponds to Last-Modified. Not writable: Although it is persisted, this header cannot be supplied on any non-admin requests.	All		X	R	P
Composite-Content-MD5	Provides an end-to-end integrity check of the content (excluding metadata) of a parallel write request at the time of completion. The supplied value is a base-64 encoding of the concatenation of the binary Content-MD5 hashes of the parts (in order) followed by a dash, followed by the number of parts as text value. On the complete request, Swarm computes the value for the overall request and reject the request with a 409 error if the values do not match.	POST	X	X	W	P
Content-MD5	Users can compute the md5 sum of the object prior to write. The header is preserved and checked on GET. A persisted Content-MD5 can also be generated by Swarm.	POST, PUT, GET, HEAD	X	X	W	P
Content-type: application/castorcontext	Required to create a context. Content-Type is a standard HTTP header.	POST, PUT	X			

Content-UUID	One header per UUID specified. Indicates the alias or primary uuid from a diskless countrep HEAD request. Also on a write request indicating the new content uuid. Unnamed objects only. COPY and PUT are invalid methods for immutable objects. Not writable: Although it is persisted, this header cannot be supplied on any non-admin requests. Is returned as a trailing header on multipart write completes.	All		X	W	P
Feed-{id}-Status	On a verbose GET or HEAD, returns the status of each defined feed, one per header. The {id} is the ID field from the feed definition. The value is 0 for success (no writes remaining), 1 for timeout, or else the number of attempts made to write the object.	GET, HEAD		X V		
Feed-{id}-StatusTime	On a verbose GET or HEAD, returns the status time of each defined feed, one per header. Time is HTTP time.	GET, HEAD		X V		
Lifepoint	Time-based SCSP and HP directives. Returned for an object created with a lifepoint header.	All	X	X	W	P
Manifest	Indicates the object is erasure coded. Indicates the manifest type of the object being created or queried. Internal. Used with replica transfer. Currently, "ec" is the only value used. Indicates the manifest type of the object being created or queried. Also returned on a verbose GET/HEAD for EC objects.	All		X V		
Node-Status	Return on a cluster status request. Indicates the current node status.			X		
Overlay-IPs	A list of space-separated IP addresses associated with the overlay key. Only on a countreps HEAD request. Up to two IP headers may appear.			X		
Overlay-Key	An overlay index key used for this object. Only on a countreps HEAD request. Up to two keys may appear.			X		
Policy-*	Include Policy-* headers in the header inclusion list or use the preserve query argument for COPY.	COPY			W	P
Policy-ECencoding	Optional; context objects only. Stores the encoding policy for erasure coding named objects in this context. Valid values: unspecified, disabled, k:p (a tuple such as 5:2 that specifies the data and parity encoding to use). For a domain, appending "anchored" cancels any policies on its buckets.	All	X	X	W	P
Policy-ECencoding-Unnamed	Optional; domain objects only. Stores the encoding policy for erasure coding unnamed objects tenanted in this domain. Valid values: unspecified, disabled, k:p (a tuple such as 5:2 that specifies the data and parity encoding to use).	All	X	X	W	P

Policy-EcMinStreamSize	Optional; context objects only. Stores the minimum object size that triggers erasure coding of named objects in this context. In units of megabytes (MB) or gigabytes (GB); must be 1MB (default) or greater. For a domain, appending "anchored" cancels any policies on its buckets.	All	X	X	W	P
Policy-EcMinStreamSize-Unnamed	Optional; domain objects only. Stores the minimum object size that triggers erasure coding of unnamed objects tenanted in this domain. In units of megabytes (MB) or gigabytes (GB); must be 1MB (default) or greater.	All	X	X	W	P
Policy-Replicas	Optional; context objects only. Stores the min, max, and/or default number of replicas that the cluster maintains for the objects in this context. For a domain, appending "anchored" cancels any policies on its buckets.	All	X	X	W	P
Policy-{Feature}-Evaluated	Evaluated policy literals, but only on contexts and the cluster status page. This header appears on GET or HEAD requests of context objects. To view it for all objects, add the "verbose" query argument.	GET, HEAD		X V		
Policy-{Feature}-Unnamed-Evaluated	Evaluated policy literals for domain objects only, related to unnamed objects therein. This header appears on GET or HEAD requests of domain objects.	GET, HEAD		X		
Policy-{Feature}-Unnamed-Evaluated-Constrained	Reports what existing constraints affect the policy evaluation for unnamed objects in a domain. If "no", nothing constrains the policy at this level. If "yes", something (anchor, override, conflict) is canceling the policy at this level. Values such as "min>2" summarize the constraints in force. This header appears on GET or HEAD requests of domain objects.	GET, HEAD		X		
Policy-{Feature}-Value-Constrained	Reports what existing constraints affect the policy evaluation. If "no", nothing constrains the policy at this level. If "yes", something (anchor, override, conflict) is canceling the policy at this level. Values such as "min>2" summarize the constraints in force. This header appears on GET or HEAD requests of context objects. To view it for all objects, add the "verbose" query argument.	GET, HEAD		X V		
Policy-Versioning	Optional; context objects only. Stores the versioning policy for objects in this context. Valid states: disabled, enabled, suspended, required.	All	X	X	W	P
Replica-Count	Returns the number of replicas found with a countreps HEAD. Returns the number of replicas created on a ROW or RmOW write. One header per UUID with a diskless countreps HEAD. Specifies the number of known replicas of the object in the cluster. Is returned only if the countreps=yes query argument is included in the request. Not writable: Although it is persisted, this header cannot be supplied on any non-admin requests.	HEAD		X	W	P

ScspHoldBucket	The SCSP hold bucket on a hold request.	HOLD		X		
ScspHoldDomain	The SCSP hold domain on a hold request	HOLD		X		
Volume	Indicates the volume UUID where one or more replicas is stored. Used with replica transfer. On a HEAD with the countreps query argument, the ids of the volume of all replicas are individually returned as separate headers. For a replicate-on-write request, there are two sets of Location and Volume headers. Not writable: Although it is persisted, this header cannot be supplied on any non-admin requests. Indicates the volume UUID where one or more replicas is stored. The Volume headers generated usually correspond to the Location headers generated.	POST, PUT, COPY, APPEND	X	X	R	P
X-{anything}-Meta[-{anything}]	Swarm custom metadata header. Uninterpreted; meant for client customization. Custom metadata has the format X-*Meta[-*], such as X-color-Meta or X-phonehome-Meta-Castor-cluster-id. The name is case-insensitive (consistent with the HTTP/1.1 RFC). Important - Custom metadata that does not match this form (such as X-Meta-color) is not persisted.	POST, PUT, COPY	X	X	W	P
X-Castor-Copy-Source	Required for parallel write part upload that copies content from an existing object. Specifies the source object by alias, UUID, or name (not Etag). Names are in the form "bucket/object."	POST	X			
X-Castor-Copy-Source-Range	Optional for parallel write part upload as a copy, failing the request if it cannot be read from the specified range. Given as bytes in the form of first - last.	POST	X			
X-Castor-Copy-Source-Domain	Optional for parallel write part upload as a copy. Supplies the domain for a named or tenanted object, failing the request if it does not exist.	POST	X			
X-Castor-Copy-Source-If-Match	Optional for parallel write part upload as a copy, returning 304 or 412 if the condition is not met. Value is an ETag (enclosed in quotes).	POST	X			
X-Castor-Copy-Source-If-Modified-Since	Optional for parallel write part upload as a copy, returning 304 or 412 if the condition is not met. Timestamp in the format of the Last-Modified header.	POST	X			
X-Castor-Copy-Source-If-None-Match	Optional for parallel write part upload as a copy, returning 304 or 412 if the condition is not met. Value is an ETag (enclosed in quotes).	POST	X			
X-Castor-Copy-Source-If-Unmodified-Since	Optional for parallel write part upload as a copy, returning 304 or 412 if the condition is not met. Timestamp in the format of the Last-Modified header.	POST	X			
X-Timestamp	On a bucket list request, the value of the Castor-System-Created header.	GET		X		

Standard HTTP Headers

Header	RFC Section	Methods	Rq	Rs	W	P	Description
Accept	7231 5.3.2						Not used by Swarm Storage.
Accept-Charset	7231 5.3.3		X				Specifies which character encodings (charsets) are acceptable for the response. Used in the console to resolve client requests.
Accept-Encoding	7231 5.3.4		X				Implemented in accordance with RFC-2616 .
Accept-Language	7231 5.3.5		X				Specifies which natural languages are acceptable for the response. Used in the console to resolve client requests.
Accept-Ranges	7233 2.3						Not used by Swarm Storage.
Age	7234 5.1	GET, HEAD		X			The age in seconds of an item read from cache. Also sent on the status page (Age: 0). Returned when an object is served from the Swarm content cache. If Age is absent, indicates the object was retrieved from disk.
Allow	7231 7.4.1	All	X	X	W	P	<p>For a POST request on an object, restricts subsequent access to the object. For a GET request, returns the list of allowed methods for that object.</p> <p>Delete protection – Allow headers have <i>no effect</i> on automatic deletes specified in Lifepoint headers. Use <code>deletable=no</code> lifepoints for best protection from deletes. Using lifepoints allows blocking recursive deletes when a bucket or domain is deleted, causing Swarm to log a CRITICAL error that non-deletable content is present.</p>
Authorization	7235 4.2		X				Answer an authorization challenge. Usually a repeated request after receiving a 401 (Unauthorized).
Cache-Control	7234 5.2	GET, HEAD, POST	X	X	W	P	Caching-related header . Specifies directives to be obeyed by all caching mechanisms along the request/response chain. Values include max-age, no-cache, and no-cache-context. Returned as metadata. Exactly matches what was sent with the object on POST. Serves as both a request header and a persisted header, which Specifies whether readers of the object access the object from cache.
Connection	7230 6.1	All	X	X			An action to perform on the connection. Includes "close", to request the client close the connection. The Swarm software implements HTTP/1.1 persistent connections. A client application is not required to close the socket /connection after each request. This header may also be absent.
Content-Encoding	7231 3.1.2.2	All		X	W	P	Activates decoding behavior. Uses registered decoders for decoding.
Content-Language	7231 3.1.3.2	All			W	P	Uninterpreted. Specifies the language(s) of the entity.

Content-Length	7230 3.3.2	All	X	X	W	P	<p>The exact number of bytes (possibly zero) comprising the content contained in the message body. Exception: If adding <code>checkIntegrity</code> to HEAD and GET requests for the same object, different Content-Length values are returned in the responses. This occurs because the HEAD response returns the Content-Length of the manifest rather than the object. This is a standard HTTP header used on all requests.</p> <p>Swarm does not update Content-Length on a COPY, and it fails any COPY requests with a Content-Length > 0. Do not send this header on a COPY unless it has the value 0.</p>
Content-Location	7231 3.1.4.2				W	P	Not used by Swarm Storage.
Content-MD5	2616 14.15	POST, PUT		X	W	P	<p>An MD5 hash of the content of the object. Can be set on a write operation or request it be computed. This value may be synthesized for some GET requests if it is not part of the persisted headers.</p> <p>This header provides an end-to-end message integrity check of the content (excluding metadata) as it is sent to and returned from Swarm. A proxy or client can check this header to detect accidental modification of the entity-body in transit. A client can provide this header to indicate Swarm needs to compute and check it as it is storing or returning the object data.</p> <p>Swarm fails a COPY request with Content-MD5 if the object is stored erasure-coded and the Content-MD5 was not stored in the original POST or PUT. Do not add Content-MD5 with a COPY to an EC object; include it in COPY requests for both erasure-coded and non-erasure-coded objects if it already existed on the object.</p> <p>S3 compatibility – The Swarm setting scsp.autoContentMD5Computation improves S3 compatibility by automating Content-MD5 hashing. The <code>gencontentmd5</code> query argument or the deprecated <code>Expect: Content-MD5</code> header on writes does not need to be included (although a separate Content-MD5 header for content integrity checking may want to be supplied). This setting is ignored wherever it is invalid, such as on a multipart initiate/complete or an EC APPEND. (v9.1)</p>
Content-Range	7233 4.2			X			Sent with a partial entity-body to specify where in the full entity-body the partial body should be applied. Appears on read range responses to indicate the actual ranges returned.

Content-Type	7231 3.1.1.5	All		X	W	P	<p>Media type as specified in the corresponding POST or PUT request. The value should be a valid media type registered with the IANA, but Swarm does not verify this or make assumptions about the content type or structure. This response can include other headers that contain meta-information supplied by the application that stored the content. In addition to a persisted content type, this value may appear on read range responses to indicate a multi-part response.</p> <p>castorcontext – <code>Content-type: application/castorcontext</code> specifies that the object is a context (domain or bucket). If the setting scsp.requireExplicitContextCreate is enabled (recommended), Swarm does not create a context object unless it includes the required header, which protects against erroneous context creation. (v9.1)</p>
Cookie	2109 4.3.4						Not used by Swarm Storage.
Date	7231 7.1.1.2	All	X	X			The HTTP time of the message. The current date/time on the Swarm node at the time of the request.
Destination	2518 9.3						Not used by Swarm Storage.
ETag	7231 2.3	All		X	R	P	Caching-related header . The ETag (entity tag) of the specific variant of the object. The value is a double-quoted UUID. The ETag of an immutable unnamed object does not change during the entire lifecycle of the object, whereas alias object ETags change each time the object is mutated by a PUT. When Versioning is enabled, the ETag identifies the version of the object. Not writable: Although it is persisted, this header cannot be supplied on any non-admin requests.
Expect	7231 5.1.1		X				Indicates that particular server behaviors are required by the client. Values include <code>100continue</code> and <code>ContentMD5</code> (unsupported by Gateway and unneeded if <code>scsp.autoContentMD5Computation</code> is enabled; see Content-MD5 Checksums).
Expires	7234 5.3	GET, HEAD		X	W	P	Caching-related header . Specifies the date/time after which the response is considered stale, for caching purposes. Uninterpreted. Returned as metadata. Matches what was sent with the object on POST.
From	7231 5.5.2						Not used by Swarm Storage.
Host	7230 5.4						The Host header field in a request provides the host and port information from the target URI, enabling the origin server to distinguish among resources while servicing requests for multiple host names on a single IP address. Swarm uses the Host header in many cases as a means of specifying the domain of the request.
If-Match	7232 3.1	All	X				Caching-related header . Used with a method to make it conditional.

If-Modified-Since	7232 3.3	GET, HEAD	X				<p>Caching-related header. Cache coherency headers.</p> <p>Per the RFC, Swarm makes no attempt to enforce "If-Modified-Since" on DELETE, PUT, or COPY requests. (v9.2)</p>
If-None-Match	7232 3.2	All	X				<p>Caching-related header. Cache coherency headers.</p> <p><i>Note:</i> If-None-Match:* can erroneously report an object exists during the time window after it is flagged for deletion by policy but before it is removed from disk. This window is determined by the HP cycle time.</p>
If-Range	7233 3.2	GET	X				<p>Caching-related header. Cache coherency headers.</p>
If-Unmodified-Since	7232 3.4	GET, PUT, DELETE	X				<p>Caching-related header. Cache coherency headers.</p>
Last-Modified	7232 2.2	All but POST		X	R	P	<p>Caching-related header. Exactly the same as Castor-System-Created. Not writable: Although it is persisted, this header cannot be supplied on any non-admin requests.</p>
Link	5988 5						Not used by Swarm Storage.
Location	7231 7.1.2			X			Values indicate how to access one or more replicas of the object directly. May be multi-valued indicating the locations of multiple new replicas.
Max-Forwards	7231 5.1.2						Not used by Swarm Storage.
Pragma	7234 5.4						Not used by Swarm Storage.
Proxy-Authenticate	7235 4.3						Not used by Swarm Storage.
Proxy-Authorization	7235 4.4						Not used by Swarm Storage.
Range	7233 3.1		X				Indicates a range of data is requested.
Referer	7231 5.5.2						Not used by Swarm Storage.
Retry-After	7231 7.1.3						Not used by Swarm Storage.
Server	7231 7.4.2	All		X			Swarm software version running on the responding node. The server name and version. CASTor Cluster/{version}.
Set-Cookie	2109 4.2.2						Not used by Swarm Storage.
TE	7230 4.3						Not used by Swarm Storage.
Trailer	7230 4.4		X	X			Indicates a trailer is sent during chunked transfer encoding.
Transfer-Encoding: chunked	7230 3.3.1	POST, PUT, APPEND	X	X			Standard header that indicates a large object to be sent to the cluster using chunked transfer encoding. Indicates that the data is being sent with an alternate transfer encoding. Values include "chunked" and "bundle". The latter is used internally for FVR of small objects (non-Standard).
Upgrade	7230 6.7						Not used by Swarm Storage.
User-Agent	7231 5.5.3		X				Standard HTTP header for a client to identify itself.
Vary	7231 7.1.4						Not used by Swarm Storage.

Via	7230 5.7.1						Not used by Swarm Storage.
Warning	7234 5.5						Not used by Swarm Storage.
WWW-Authenticate	7235 4.1			X			Indicates an authentication challenge. Usually associated with a 401 (Unauthorized) response.

Filtering Headers

A business requirement may exist to enable filtering of the optional HTTP response headers transmitted for GET and HEAD requests if Swarm is used to deliver content directly over the Internet. (v9.5)

Caution

Indiscriminate filtering of response headers, which is cluster-wide in scope, can break client applications. Do not filter headers if the client applications are object storage aware and are using SCSP or S3 (Content Gateway) to interact with Storage.

Filtering metadata headers on objects can cause problems for other applications that know how to work with object metadata, such as Content UI, SwarmFS, and FileFly.

Because the header filtering does add additional processing to Swarm's responses, best practice is to enable it only for a specific content delivery need:

- Bandwidth needs to be conserved and as many bytes as possible need to be eliminated when serving content.
- Enhanced security is needed and as little as possible about content and context needs to be revealed.
- The target clients are web browsers instead of object storage aware applications.

Important

Regardless of filtering, do not expose Swarm Storage directly on the Internet. Do not allow arbitrary requests, especially by unauthorized users. Some kind of HTTP request restrictions should always be present to prevent abuse by untrusted clients.

Header filtering is a Storage feature dynamically implemented without a cluster restart. The choice of filtering approaches follow:

- **Whitelist** – list which non-required headers to retain, if any
- **Blacklist** – list which non-required headers to remove, preserving all others

The lists are case-insensitive, and they can include system headers (such as "Castor-System-Owner").

Essential headers

The following essential metadata headers are unaffected by Blacklisting and are always included when they are present on an object:

Allow, Authentication-Info, Authorization, Cache-Control, Connection, Content-Length, Content-MD5, Content-Range, Content-Type, Date, Expires, Keep-Alive, Location, Server, Trailer, Transfer-Encoding

Settings for Filtering

Filtering is disabled by default. These SCSP settings allow controlling which of the optional response headers are returned from the cluster:

scsp. filterResponseHeaders	none	Which method to use to filter HTTP response headers. Whitelist or blacklist setting must be defined before implementing that method. Valid values: none, whitelist, blacklist. SNMP: filterResponseHeaders
scsp. filterResponseBlacklist	[]	Which headers to remove from HTTP GET and HEAD responses. List is comma-separated and case-insensitive. SNMP: filterResponseBlacklist
scsp. filterResponseWhitelist	[]	Which headers to retain in HTTP GET and HEAD responses, removing all others. List is comma-separated and case-insensitive. Leave the brackets empty to have Swarm strip out all non-essential headers. SNMP: filterResponseWhitelist

Best practice

To avoid a window when filtering is enabled but the filter list is empty, define the whitelist or blacklist *first* and then enable filtering by setting `scsp.filterResponseHeaders`.

Set these values using the [Storage UI](#), or use SNMP or cURL:

```
curl -i http://$SCSP_HOST:91/api/storage/clusters/<cluster-name>/settings/scsp.filterResponseWhit
-XPUT -d {"value": ["key1", "key2"]}
```

```
curl -i http://$SCSP_HOST:91/api/storage/clusters/<cluster-name>/settings/scsp.filterResponseHead
-XPUT -d {"value": "whitelist"}
```

Sample Output

Following are examples of how responses can appear with and without filtering applied. Swarm includes the `Castor-System-Headers-Filtered: True` header with every response that has been filtered by a whitelist or blacklist.

Target of GET	Headers Not Filtered	Headers Filtered
---------------	----------------------	------------------

<p>Named Object</p>	<pre>\$ curl -i "172.16.15.180/bucket/stream?domain=domain" -I HTTP/1.1 200 OK Castor-System-CID: 84c1cbf7d33aec1feec4d4dd11225b87 Castor-System-Cluster: CASTorCluster Castor-System-Created: Fri, 30 Nov 2018 16:33:44 GMT Castor-System-Name: stream Castor-System-Version: 1543595624.202 Content-Length: 0 Last-Modified: Fri, 30 Nov 2018 16:33:44 GMT Etag: "46ce386cdc13828d7d8d68ee20aac58d" Castor-System-Path: /domain/bucket/stream Castor-System-Domain: domain Volume: 0a9a7ed07b5f86520b096fb0ef824846 Date: Fri, 30 Nov 2018 16:34:21 GMT Server: CASTor Cluster/9.6.a Keep-Alive: timeout=14400</pre>	<pre>\$ curl -i "172.16.15.179/bucket/s?domain=x" HTTP/1.1 200 OK Content-Length: 0 Castor-System-Headers-Filtered: True Date: Fri, 30 Nov 2018 16:33:48 GMT Server: CASTor Cluster/9.6.singleip Keep-Alive: timeout=14400</pre>
---------------------	--	--

Undefined Responses from Swarm

- [Critical error in Swarm](#)
- [Unsupported request](#)
- [Unavailable service](#)
- [Unsupported HTTP version](#)

This section describes HTTP requests sent to a storage cluster where the results are currently undefined. In most cases, one of the following error responses is sent by Swarm.

Critical error in Swarm

```
HTTP/1.1 500 Internal Server Error
Date: Wed, 1 Sept 2012 15:59:02 GMT
Server: CASTor Cluster 5.0.0
Allow: GET, HEAD, POST, PUT, DELETE
Connection: close
Content-Length: 27
Content-Type: text/html CRLF
Message
```

Check your logs for more information and contact your support representative if necessary.

Unsupported request

```
HTTP/1.1 501 Not Implemented
Date: Wed, 1 Sept 2012 15:59:02 GMT
Server: CASTor Cluster 5.0.0
Allow: GET, HEAD, POST, PUT, DELETE
Connection: close
Content-Length: 56
Content-Type: text/html CRLF
Swarm does not understand the request or does not yet implement this functionality.
```

This response indicates Swarm received a request method not implemented. The methods listed in the Allow header currently work in Swarm.

See [SCSP Headers](#).

Unavailable service

```
HTTP/1.1 503 Service Unavailable
Date: Wed, 1 Sept 2012 15:59:02 GMT
Server: CASTor Cluster 5.0.0
Allow: GET, HEAD, POST, PUT, DELETE
Connection: close
Content-Length: 0
Content-Type: text/html
```

This response indicates Swarm received a request it did not understand or it does not have the resources to process the request. The client should resubmit the request at a later time or to a different node in the cluster.

Unsupported HTTP version

```
HTTP/1.1 505 HTTP Version not supported
Date: Wed, 1 Sept 2012 15:59:02 GMT
Server: CASTor Cluster 5.0.0
Allow: GET, HEAD, POST, PUT, DELETE
Connection: close
Content-Length: 0
Content-Type: text/html
```

This response indicates a request was received with an HTTP version other than HTTP/1.1. Swarm only supports HTTP/1.1.

Formatting SCSP Commands

- [Important](#)
- [When to include domain and Host](#)
- [Calling named objects](#)
- [Calling unnamed objects](#)
 - [Important](#)
 - [Caution](#)



Important

All commands include specific formats for named objects and for unnamed objects.

When to include domain and Host

Domain is required is for SCSP methods on a domain object itself. Neither `domain` nor `Host` is required for requests *within* the default cluster domain; otherwise, the domain name must be passed as the `Host` in the request. A cluster needs to have one domain with the same name as the cluster, which sets up a default cluster domain.

Client applications most often send the domain name as the `Host` in the request. The client can supply the `domain` argument to explicitly override any value from the `Host` request header when the `Host` header does not match the domain name, . A `domain` argument always has precedence over the `Host` header in the HTTP/1.1 request.

Calling named objects

The named object format is:

```
METHOD /bucketname/objectname[?query-arguments] HTTP/1.1
```

where

- **bucketname** is a URL-encoded identifier that *cannot* contain slash characters or any other character not allowed in HTTP URL
- **objectname** is any legitimate URL, which *can* contain slash characters.

Calling unnamed objects

The unnamed object format is:

```
METHOD /[uuid][?query-arguments] HTTP/1.1
```

Specify the UUID with all SCSP methods *except* WRITE, in which case the cluster returns the UUID in the response if the write is successful.

Important

Use a **HOST** header equivalent to the cluster name, the host IP address, or a **domain=clusterName** query argument on all requests *even if not using domains for other purposes* when writing unnamed objects.

Caution

Verify the application is not passing a **HOST** header equal to neither an IP address nor a domain that exists in the cluster unless the host header matches the cluster name when writing unnamed objects. Swarm attempts to look up the non-existent domain on every request and waits for multiple retries before the lookup times out, impacting performance.

SCSP Compatibility and Support

- [Determining the Swarm and SCSP proxy version](#)
 - [Using a browser](#)
 - [Using the node IP address or Host name](#)
 - [Using the SCSP proxy external IP address or Host name](#)
- [Issues with 100-Continue Header](#)
 - [Best practice for integrators](#)

This section lists major API-level features and changes in Swarm releases starting with version 4.0 to assist with writing client applications.

Determining the Swarm and SCSP proxy version

To determine which version of Swarm is running on a node, search for the `Server` header in any response by:

- Using a web browser.
- Sending a GET / request to the node using the node IP address or host name.
- Sending a GET / request to the node using the SCSP Proxy's external IP address or host name.

Using a browser

To search for the `Server` header using a web browser, use a browser with a head capture utility (such as Live HTTP Headers with Firefox) and enter the following URL in the `Address` field:

```
http://node-ip\[ :scsp-port\]
```

where `scsp-port` is required only if using a value other than the default value of 80.

Using the node IP address or Host name

Send a GET / request to the node using the node's IP address or host name as the `Host` in the request if a client application is on the same subnet as a Swarm node.

In this example, the responding node is running Swarm version 5.1.

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 733
Cache-Control: no-cache
Expires: Thu, 03 Jun 2011 19:09:05 GMT
Age: 0
Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE
Castor-System-TotalGBAvailable: 145
Castor-System-TotalGBCapacity: 156
Castor-System-Cluster: cluster.example.com
Etag: "8c2c582c216a1f088c3652bced5a5f91"
Date: Fri, 04 Mar 2011 22:55:45 GMT
Server: CASTor Cluster/5.1.0
```

Using the SCSP proxy external IP address or Host name

Send a GET / request to the Swarm SCSP Proxy, using the SCSP Proxy's external IP address or host name as the `Host` in the request if a client application is not on the same subnet as a Swarm node.

In this example, the SCSP Proxy is running version 1.4 and Swarm is running version 6.0.

```
HTTP/1.1 200 OK
Scsp-Proxy-Cluster: cluster.example.com
Content-Type: text/plain
Content-Length: 0
Cache-Control: no-cache
Expires: Tue, 13 Sep 2011 11:06:36 GMT
Castor-System-TotalGBAvailable: 148567
Castor-System-TotalGBCapacity: 349123
Scsp-Proxy-Nodes: count=16
SCSP-Proxy-Agent: SCSP Proxy Service/1.4.0
Age: 0
Etag: "6a04a4fef71925b92ec12de887ac4653"
Via: 1.1 myhost.example.com (SCSP Proxy Service/1.4.0)
X-Forwarded-For: myhost.example.com
X-Forwarded-Server: myhost.example.com
Date: Wed, 14 Sep 2011 14:53:16 GMT
Server: CASTor Cluster/6.0.0
```

Issues with 100-Continue Header

Inconsistencies appear when the Swarm SDK is not integrated with the 100 Continue status header to implement the SCSP protocol.



Best practice for integrators

Use the [Swarm SDK](#), which includes full implementations of the SCSP protocol in multiple languages.

Consider the following inconsistencies with 100-Continue for integrators not using the SDK:

- **Python httplib.** The [Python httplib](#) wrapper behavior is not altered in the presence of a 100-continue header and sends the complete request body without waiting for the continue response from the server. The Python SDK does not use httplib and does handle 100-continue headers correctly.
- **C#/.NET WebClient/HttpRequest.** [HttpRequest](#) behavior is not altered in the presence of a 100-continue header and sends the complete request body without waiting for the continue response from the server. Client applications are informed when encountering a 100-continue header.
- **Java Apache Commons HTTP client.** The [Apache commons HTTP client](#) does handle 100-continue correctly after setting the POST method parameter:

```
method.getParams().setParameter(HttpMethodParams.USE_EXPECT_CONTINUE, new Boolean(true));
```

See [RFC 7231 section 6.2.1](#) for more about the 100 Continue status.

Error Response Headers

SCSP error responses have headers detailing the error code, response code (token), and error description. The response for any request with an error code 400 or greater includes three special headers: `castor-system-error-code`, `castor-system-error-token`, and `castor-system-error-text`. These headers replace the legacy header `x-castor-meta-error-message`, which is deprecated. (v9.1)

Tip

A failure response may contain a series of cascading errors. Focus on the *final* error in the series, which is likely to be the issue needing to be addressed:

```
< HTTP/1.1 412 Precondition Failed
< Castor-System-Error-Token: SecurityRealmFailure
< Castor-System-Error-Text: Failed to load context 'some-domain.example.com/nosuchbucket'.
< Castor-System-Error-Code: 404
< Castor-System-Error-Token: RequiresContext2
< Castor-System-Error-Text: Cannot find required domain or bucket.
< Castor-System-Error-Code: 412
< Content-Length: 130
< Content-Type: text/html
...

```

- A **text** value of "{0}" or "{1}" indicates a variable to be populated by Swarm when the value is generated.
- A **code** value of "0" indicates the response code is not static, and is populated by Swarm when the error is generated.

Code	Token	Text
0	CloseException	{0}
0	CloseException2	{0}
0	CloseException3	{0}
500	CloseFailure	{0}
500	CloseFailure2	{0}
500	CloseFailure3	{0}
500	CloseFailure4	{0}
507	ClusterOutOfObjects	Not enough licensed objects in the cluster for request.
507	ClusterOutOfSpace	Not enough licensed space in the cluster for request.
507	ClusterOutOfSpace3	Not enough space in cluster.
507	ClusterOutOfSpace4	Not enough space in cluster.
0	CompletionErrorNonRequest	{0}

0	CompletionErrorRequest	{0}
500	CompletionErrorUnknown	Unknown completion error.
412	ConditionalIfMatch	If-Match condition not met.
412	ConditionalIfMatch2	If-Match condition not met.
304	ConditionalIfModified	If-Modified-Since condition not met.
304	ConditionalIfNoneMatch	If-None-Match condition not met on GET or HEAD.
412	ConditionalIfNoneMatch2	If-None-Match condition not met.
412	ConditionalIfUnmodified	If-Unmodified-Since condition not met.
412	ConditionalMatchNotFound	A matching object was not found on If-Match request.
412	ConditionalNoneMatchFound	A matching object was found on If-None-Match request.
400	CopyMD5Mismatch	The Content-MD5 provided on the COPY request does not match the value in the manifest.
400	DigesterForbiddenArgs	Content Integrity: 'hashtype' and 'newhashtype' queryArgs cannot be used together.
400	DigesterInvalidHashtype	Content Integrity: unsupported hash type.
400	DigesterMismatch	Persisted {0} did not match request.
400	DigesterMismatch2	{0} did not match computed digest.
400	DigesterMismatch3	Local Content-MD5 did not match remote Content-MD5.
400	DigesterMismatch4	{0} did not match computed digest.
400	DigesterMissingHeader	Expect {0} trailing header not supplied.
500	DigesterMissingRemote	Remote replica did not contain a Content-MD5.
400	DigesterRequiresHash	Content Integrity - 'hash' query arg required with 'hashtype' on GET or COPY.
400	DigesterRequiresHashtype	Content Integrity - 'hashtype' query arg required on request.
412	DigesterSealMismatch	Content Integrity - seal did not validate.
503	ECSegmentClose	Premature segment close on read. Please retry.
406	EncodeInvalidCE	Content-encoding not acceptable.
405	EncodeInvalidMethod	APPEND not allowed for content-encoded objects.
406	EncodeRequiresCE	Content-encoding not found on request or in 'decoderSettings' setting.
500	EnvelopedHeadersTimeout	Content-Type: %s used, but enveloped headers were not sent before timeout.
400	FeedSendHeaderMisuse1	%s header should not be provided on feed SEND request.

400	FeedSendHeaderMisuse2	%s header should not be provided on feed SEND request.
400	FeedSendHeaderMisuse3	%s header should not be provided on feed SEND request.
400	FeedSendIllegalFeedId	feedid query argument must refer to the id of a feed
400	FeedSendIllegalFeedType	feedtype query argument must refer to a feed type.
400	FeedSendIllegalTimeout	timeout query argument must be a positive number.
400	ForbiddenAction	'Action' query arg is not allowed on existing object request.
400	ForbiddenAliasRename	An alias stream cannot be renamed.
400	ForbiddenAliasUUID	Only the COPY operation supports renaming using aliasuuid.
400	ForbiddenConditional	Conditional headers other than If-None-Match:* on a named request not allowed in a POST.
417	ForbiddenContentMD5	Expect:Content-MD5/gencontentmd5 is not supported for APPEND.
400	ForbiddenContext	Cannot write duplicate context.
409	ForbiddenContextName	POST will only create new context objects. Context already exists.
409	ForbiddenContextName2	POST will only create new context objects. Context already exists.
412	ForbiddenDomainName	Cannot rename domain to existing domain name.
400	ForbiddenEtag	Etag query argument not appropriate on write requests.
501	ForbiddenFeature	You must configure Erasure Coding to obtain this functionality.
501	ForbiddenFeature2	You must configure Erasure Coding to obtain this functionality.
400	ForbiddenGenID	IsGenId query arg must be used with GET or HEAD methods only.
400	ForbiddenIndexWaitValue	Forbidden value '{0}' with index query argument
400	ForbiddenManifest	{0} not appropriate on this request.
400	ForbiddenManifestEC	Cannot provide erasure coding query args for a manifest write.
400	ForbiddenManifestHeader	'{0}' header only allowed on a POST.
400	ForbiddenManifestHeader2	'{0}' header not allowed on a context request.
405	ForbiddenMethod	Allow header forbids this method on this object.
409	ForbiddenMutable	Operation on mutable object must be by name or alias.
409	ForbiddenMutable2	Can not PUT an immutable object.
409	ForbiddenMutable3	Can not PUT an immutable object.
409	ForbiddenMutable4	Can not COPY an immutable object.
409	ForbiddenMutable5	Can not APPEND an immutable object.

409	ForbiddenMutable6	Can not {0} an immutable object.
400	ForbiddenPolicyHeader	Policy headers are only allowed on domains and buckets.
400	ForbiddenPutCreate	'Putcreate' query argument not allowed on context requests.
412	ForbiddenRename	Cannot rename due to existing object.
400	ForbiddenSegmentEC	Cannot EC an existing segment.
400	ForbiddenSegmentSize	Cannot specify 'segmentsize' on an existing EC object.
409	ForbiddenSegmented	Cannot segment an EC object.
409	ForbiddenSegmented2	Cannot segment an EC object.
400	ForbiddenSegmentedEC	Cannot specify segmented=yes, and provide erasure coding query args.
400	ForbiddenSpec	Cannot specify 'erasurecoded' on an existing object for given method.
400	ForbiddenSpec2	Cannot specify 'encoding' on an existing object for given method.
400	ForbiddenSpec3	Cannot specify 'segmentwidth' on an existing object for given method.
400	ForbiddenSpec4	Cannot specify 'segmentsize' on an existing object for given method.
400	ForbiddenSpec5	Cannot specify 'lifepoint k:p' on an existing object for given method.
400	ForbiddenStreamHeader	Duplicate header values detected for {0}.
403	ForbiddenUUID	UUID forbidden on POST.
403	ForbiddenUUIDName	UUID/Name forbidden on RETRIEVE.
400	ForbiddenVersioning	'Version' query argument may not be used on a context request.
400	ForbiddenVersioning2	'Version' query argument may not be used on an immutable request.
400	ForbiddenVersioning3	DEPRECATED: 'Version' query argument not appropriate in current state.
409	GenRequiresEncoding	GEN can only be performed on segments, by administrative request.
412	IfNoneMatchFail	Named stream already exists with If-None-Match: * specified.
503	InsufficientMemory	Insufficient physical memory
503	InsufficientMemory2	Service unavailable, busy
507	InternodeInsufficientResources	Could not find sufficient resources.
507	InternodeInvalidDistribution	Did not balance across the correct number of subclusters.
507	InternodeInvalidDistribution2	Did not balance evenly across subclusters.
507	InternodeInvalidDistribution3	Did not balance evenly across nodes.
507	InternodeInvalidDistribution4	Did not balance evenly across subclusters.

503	InternodeInvalidState	Failed to detect valid subcluster bids. Try again.
507	InternodeRequiresNodes	No remote nodes are available to take objects of specified size.
507	InternodeRequiresNodes2	Not enough remote nodes are available to take objects of specified size.
507	InternodeRequiresNodes3	Not enough subclusters are available to take objects of specified size.
507	InternodeRequiresNodes4	No nodes found in bid auction.
507	InternodeRequiresSubclusters	Cannot apply subcluster protection when the segment count is not greater than the required segments per subcluster.
500	InternodeUnexpectedCount	Found an unexpected number of nodes, fewer than needed.
507	InternodeUnexpectedCount2	Unexpected number of volumes.
400	InvalidAliasUUID	'Aliasuuid' must be a UUID.
400	InvalidAuthorization	CASStor-authorization header error.
410	InvalidBasisStream	Basis object has been deleted.
503	InvalidBirthdate	Temporary alias conflict after upgrade: try again.
400	InvalidBucketName	Illegal character in bucket name.
400	InvalidCID	'Cid' queryArg must be valid UUID.
400	InvalidContentLength	Content-Length must be zero for COPY request.
400	InvalidContentLength1	WritePattern query argument requires a message body with contentLength greater than 0.
400	InvalidContentLength2	WritePattern pattern must not be larger than value of scsp.writePatternMax.
400	InvalidContentMD5	Content-MD5 not allowed on multipart write initiate request.
400	InvalidContentMD52	Content-MD5 value can not be blank.
400	InvalidContentMD53	Content-MD5 value was not not a valid base64 md5 hash.
400	InvalidCount	'Count' query arg must be > 0 and <= scsp.maxreplicas.
400	InvalidCount2	'Count' query arg must be an integer > 0 and <= scsp.maxreplicas.
400	InvalidDecorates	{0} header may not be added to contexts.
400	InvalidDecorates2	{0} header must be a single UUID.
400	InvalidDecorates3	{0} header must be a valid UUID.
400	InvalidDecorates4	{0} header value must refer to an ETag.
400	InvalidDomainName	Illegal character in domain name.
400	InvalidDomainSpecified	Domain may not change on a recreatecid request.

400	InvalidDomainsDomain	Request must not provide both 'domains' and 'domain' query arguments.
400	InvalidECCombination	Cannot specify no encoding, and provide erasure coding query args.
400	InvalidECEncoding	Invalid EC encoding.
400	InvalidECQueryArgs	Error parsing EC queryArgs.
400	InvalidEntityLength	Expected integer for 'Entity-Length' header value.
400	InvalidEntityLength2	Entity length header does must match Content-Length header.
417	InvalidExpectContentMD5	Expect: Content-MD5/gencontentmd5 not allowed on multipart write initiate request.
400	InvalidMethod	A PATCH request is only valid on a multipart upload initiate.
400	InvalidMethodForListing	A listing request with the 'format' query arg must be a GET not {0}.
400	InvalidModifiedSince	If-Modified-Since time in the future.
400	InvalidNewName	Domain rename name contains invalid character.
400	InvalidNewName2	Bucket or object rename name is invalid.
400	InvalidObjectName	OBSOLETE - Object names cannot look like a UUID.
500	InvalidOwner	Invalid owner header format.
400	InvalidParallelEncoding	Part uploads cannot specify encoding.
400	InvalidParallelFlag	Part uploads cannot be alias objects.
400	InvalidParallelMethod	Part upload must be a POST.
400	InvalidParallelRename	Part uploads cannot specify a new name.
400	InvalidParallelRequest	Context requests cannot be made by multipart upload.
409	InvalidParallelState	Cannot PUT a multipart write object still in progress.
409	InvalidParallelState2	Cannot COPY a multipart write object still in progress.
409	InvalidParallelState3	Cannot APPEND a multipart write object still in progress.
400	InvalidPartNumber	Part number must be an integer at least 1.
400	InvalidPolicyHeader	Unrecognized policy header.
400	InvalidPolicyHeader2	Unnamed policy cannot be used on a bucket.
400	InvalidPolicyValue	Policy value is not valid.
400	InvalidPolicyValue2	Policy value is malformed or invalid.
400	InvalidQueryArgCombo	The 'replace' and 'preserve' query args can not be used on the same request.
400	InvalidRangeTime	If-Range time in the future.

400	InvalidRecreateCID	The value of recreatecid must be a UUID.
400	InvalidRepSpec	Cannot specify 'erasurecoded' and reps=X.
400	InvalidRepSpec2	Cannot specify 'encoding' and reps=X.
400	InvalidRepSpec3	Cannot specify 'segmentwidth' and reps=X.
400	InvalidRepSpec4	Cannot specify reps=X on a chunked upload. Chunked uploads must use EC.
400	InvalidReplicate	'Replicate' query arg must be > 0 and <= scsp.maxreplicas.
400	InvalidReplicate2	'Replicate' query arg must be a keyword, or an integer > 0 and <= scsp.maxreplicas.
400	InvalidSegmentSize2	SegmentSize must be greater than or equal to value of segmented.minSegmentSize.
400	InvalidSourceHeader	{0} header value must be in the form ..
400	InvalidSourcesHeader	{0} header value must be a comma-separated list in the form ..
400	InvalidStreamHeader	Header '{0}' is not syntactically valid.
400	InvalidURI	Invalid URI. Bucket and object name must be percent-encoded utf-8 bytes.
400	InvalidURI2	URI resource does not match request.
412	InvalidURI3	Could not resolve domain for context specified in CID header.
400	InvalidURI4	Could not decode bucket name.
400	InvalidUnmodifiedSince	If-Unmodified-Since time in the future.
400	InvalidUploadID	UploadId query arg value was not a well-formed uploadid value.
409	InvalidUploadID2	Initialized object must have matching uploadID. It might have been updated since the initiate.
400	InvalidVersion	Invalid 'version' query argument value.
400	InvalidVersionContext	'Version' query argument may not be used on a context request.
400	InvalidVersionMethod	'Version' query argument may not be used with request method.
400	InvalidWritePattern	Value for writePattern must be a positive integer.
400	InvalidWriteRandom	Value for writeRandom must be a positive integer.
400	MetalInvalidUUID	There are no valid UUIDs to query for countreps.
500	MultiHeaderMismatch	Local and remote replicas are not identical.
404	NotFound3	Existing object not found in cluster.
404	NotFound4	Requested object was not found.
404	NotFound5	Requested object was not found.

503	NotFound6	Requested object was not found. Try again.
0	NotFoundDeleted	Requested object was not found because it has been deleted.
404	NotFoundDeleted2	Existing object found with delete marker.
0	NotFoundDeleted3	Requested object was deleted.
404	NotFoundDeleted4	Requested object has a delete marker.
0	NotFoundDeleted5	Requested object was deleted.
404	NotFoundDeleted6	Requested object has a delete marker.
404	NotFoundDeleted7	Requested object was deleted by policy.
404	NotFoundDeleted8	Requested object was deleted by policy.
404	NotFoundDeleted9	Requested object was deleted by policy.
0	NotFoundDeletedExisting	Existing object found deleted in the cluster.
404	NotFoundDisk	Object was not found on disk.
404	NotFoundExisting	Existing object not found in the cluster.
0	NotFoundExistingDeleted	Existing object found deleted in the cluster.
503	NotFoundExpired	Requested object was expired by policy.
503	NotFoundExpired2	Requested object was expired by policy.
503	NotFoundExpired3	Requested object was expired by policy.
503	NotFoundExpired4	Requested object was expired by policy.
412	NotFoundInitiateDeleted	Initiated object found deleted in the cluster.
404	NotFoundMarker	Versioned object has a delete marker.
404	NotFoundMarker2	Object has a delete marker.
404	NotFoundNotVersioned	Requested object was not found, not versioned.
404	NotFoundNotVersioned2	Requested object was not found, not versioned.
409	NotFoundObsolete	Object version obsolete.
409	NotFoundObsolete2	Object version present or obsolete.
404	NotFoundOld	Looking for an older version in a versioning disabled state.
404	NotFoundOld2	Looking for an older version in a versioning suspended state.
404	NotFoundOld3	Requested object's current version was not found in the cluster.
404	NotFoundOverlay	Requested object was not found, per overlay index lookup.

404	NotFoundOverlay2	Requested object was not found, per overlay index lookup.
404	NotFoundOverlay3	Requested object was not found, per overlay index lookup.
404	NotFoundOverlay4	Requested object was not found, per overlay index lookup.
404	NotFoundVersion	Requested object was not found in the cluster.
404	NotFoundWrite	Requested object was not found.
0	NotFoundWrite2	Requested object was deleted.
500	OpenFailure	{0}
500	OpenFailure2	{0}
400	OperationNotPermitted	The 'forcetrim' query arg is only permitted on a HEAD request.
400	OperationNotPermitted2	The 'forcetrim' query requires administrative authorization.
500	PipelineFailure	Can't process request
400	RangeForbidden	Range not allowed on COPY or APPEND.
400	RangeInvalid	Invalid range; index greater than range.
416	RangeInvalidHeader	Invalid range header; could not parse.
416	RangeInvalidHeader2	Invalid range header; could not parse.
500	RangeUnexpectedContent	Unexpected extra content in multipart response.
404	Reader404NotFound	The 404stream query arg was provided.
401	ReaderAuthError	Destination cluster authorization error.
400	ReaderBucketError	List operation specifies non-context object.
503	ReaderBucketError10	Indexer searches require node be fully up.
400	ReaderBucketError2	List request must not provide both 'domains' and 'domain' query arg.
400	ReaderBucketError3	List request must provide either a 'domain' or 'domains' query arg.
412	ReaderBucketError4	Supplied realm does not exist.
400	ReaderBucketError5	List request must not provide a UUID.
417	ReaderBucketError6	List operations do not support any 'Expect' headers.
501	ReaderBucketError7	List operations unavailable because indexer is not configured or not licensed.
501	ReaderBucketError8	List operations require Indexing Feed, but none currently available.
400	ReaderBucketError9	Indexer searches on selected field not supported.
404	ReaderCacheDeleted2	Requested object was found with a delete marker.

409	ReaderContextMismatch	Object context mismatch. Invalid domain or possible attempt to tamper with data.
409	ReaderContextMismatch2	Object context mismatch. Invalid domain or possible attempt to tamper with data.
409	ReaderContextMismatch3	Domain specified for untenanted object.
500	ReaderDeleteError	Disk error on open.
500	ReaderDeleteException	Exception deleting object.
412	ReaderDeleteNoExist	Failed to open object for deletion.
503	ReaderDeleteNoSpace	Not enough space in node.
401	ReaderDeleteNotAuthorized	Unauthorized administrative request.
403	ReaderDeleteNotDeletable	object is not deletable at this time, based on its lifepoint policy.
404	ReaderDeleteNotFound	Object has been deleted.
503	ReaderDeleteOutOfMemory	Not enough index memory to complete the operation. Try again.
500	ReaderDifferent	Named object lookup failure.
503	ReaderIndexError	Index error: try again.
0	ReaderIndexError2	Index error.
400	ReaderIndexError3	Malformed listing request.
409	ReaderInvalidCID	Object context mismatch. Invalid domain or possible attempt to tamper with data.
409	ReaderInvalidCID2	Object context mismatch. Domain specified for untenanted object.
400	ReaderInvalidFields	Fewer sort fields than marker values not supported.
400	ReaderInvalidFormat	Invalid format value. Must be one of: json, xml.
400	ReaderInvalidHeader	{0} header for remote cluster must be different than cluster.name setting.
400	ReaderInvalidHeader2	{0} header value should be in the form :.
400	ReaderInvalidHeader3	{0} header value should be of the form :.
400	ReaderInvalidHeaders	Error formatting object headers.
409	ReaderInvalidId	Mutable objects must be deleted by name or alias.
500	ReaderInvalidManifest	Invalid manifest.
500	ReaderInvalidManifest2	Invalid manifest.
400	ReaderInvalidRead	Destination cluster reports 400 on read request.
504	ReaderInvalidRead2	Cannot connect to destination.
0	ReaderInvalidRead3	Unexpected destination cluster response.

502	ReaderInvalidRead4	Destination server version does not support remote replication.
502	ReaderInvalidRead5	Invalid destination server version.
502	ReaderInvalidRead6	Destination server is not recognized.
502	ReaderInvalidRead7	Destination server name is not provided.
400	ReaderInvalidRecursive	'Recursive' query arg must be an integer.
400	ReaderInvalidStreamResults	Invalid streamresults value. Must be one of: true, false, yes, no.
400	ReaderInvalidStreamType	Invalid stype value. Must be one of: {0}.
500	ReaderInvalidType	Unexpected manifest object type.
400	ReaderInvalidUUID	Missing or invalid UUID.
409	ReaderInvalidUploadId	UploadID in object was not found or does not match UploadID in request.
400	ReaderInvalidVersion	{0} header not allowed on non-administrative requests.
400	ReaderInvalidVersion2	{0} header must be a float.
400	ReaderInvalidVolume	Invalid volume specified.
400	ReaderInvalidVolume2	Volume specified is not ready.
500	ReaderMissingDigest	Partial object, has no digest.
412	ReaderNoManifest	No manifest on a non-EC object.
502	ReaderNoSources	Could not find a source for RETRIEVE operation.
503	ReaderNotFound2	Requested object was not found on disk: try again.
404	ReaderNotFound3	Attempting to read a version not associated with this nid.
404	ReaderNotFound5	Attempting to read a version not associated with this alias.
500	ReaderNotFound6	Alias object lookup failure.
503	ReaderNotFound7	Object version found is obsolete; try again.
500	ReaderNotFound8	IOError opening existing object.
503	ReaderNotFound9	Object exists but is not readable; try again later.
0	ReaderNotFoundAlreadyDeleted	Requested object was not found, already deleted.
404	ReaderNotFoundCacheDeleted	Requested object was found with a delete marker.
404	ReaderNotFoundPolicy	Destination object is not found, is deleted by policy.
404	ReaderNotFoundProxy	Primary UUID not found while proxying object.
0	ReaderProxyError	Failed to read object necessary to proxy.

502	ReaderRequiresCluster	Destination cluster.name is not set.
502	ReaderRequiresClusterMatch	Destination cluster.name differs from expected.
400	ReaderRequiresHeader	SEND request requires {} header.
400	ReaderRequiresHeader2	{0} header not provided on request.
0	ReaderRequiresMarker	Failed to create delete marker in cluster.
503	ReaderRequiresMarker2	Exception trying to create delete marker in cluster.
400	ReaderRequiresName	cluster.name setting is required for SEND request.
0	ReaderRequiresRead	Unexpected read error attempting to proxy object.
500	ReaderRequiresRead2	Unexpected exception when proxying for EC manifest.
0	ReaderSegmentError	Unable to read a segment.
0	ReaderSegmentError2	Unable to info a segment.
0	ReaderSegmentError3	Unable to read a segment.
500	ReaderSegmentError4	Unknown read error.
410	ReaderSegmentError5	Not enough segments found to service request.
500	ReaderSegmentError6	Caught RequestError during ec segment prepare.
0	ReaderSegmentError7	Unable to info segment.
401	ReaderUnauthorized	Unauthorized administrative request.
401	ReaderUnauthorizedInternode	Unauthorized internode request. Segments requests require admin auth.
503	ReaderUnavailableIndex	Search index unavailable. Wait indexer.insertBatchTimeout seconds and try again or check log for indexer errors: try again.
500	ReaderUnexpectedBatch	Batch handler exception.
500	ReaderUnexpectedJournal	Could not iterate journal.
500	ReaderUnexpectedStatus	Error computing status page.
500	ReaderUnexpectedType	Unexpected object type.
400	RequiredBidLength	Either a content length header or extentsize query arg is required to create a bid.
400	RequiredLength	A request must specify a length or qualify for EC in order to estimate required license space.
400	RequiresAdmin	Domain rename requires 'admin' queryArg.
409	RequiresBasisGeneration	Generation of basis stream has changed since init.
409	RequiresBasisHeader	The initiate object does not refer to the request object.

410	RequiresBasisStreamNotFound	Could not find basis object.
400	RequiresCluster	{0} not specified on request.
411	RequiresContentLength	Content-Length not provided for non-EC request.
411	RequiresContentLength2	Content-Length must be provided and must be zero for COPY request.
400	RequiresContext	The 'recreatecid' query arg requires that either a domain or bucket is specified on the request.
412	RequiresContext2	Cannot find required domain or bucket.
400	RequiresDecorates	{0} header object was not found.
412	RequiresDomain	Request requires a domain specification.
412	RequiresDomain2	Tenancy enforced and failed to find or load domain '{0}'. If creating a domain, include the 'Content-type: application/castorcontext' header.
412	RequiresDomain3	A loadable domain must be specified when enforceTenancy=True.
400	RequiresDomain4	Part upload for tenanted write must include domain query arg.
400	RequiresEncoding	Multipart writes must specify a valid EC encoding in the cluster, or on the request.
409	RequiresEncoding2	Could not find encoding header in initiated object.
409	RequiresEncoding3	Encoding has changed on the basis since the initiate.
409	RequiresEncoding4	Could not find encoding header in basis object.
400	RequiresExplicitBucket	Bucket creation requires the 'Content-type: application/castorcontext' header.
400	RequiresExplicitContext	Context creation requires the 'Content-type: application/castorcontext' header.
507	RequiresHealthReporting	Health reporting is required by license.
410	RequiresInitiateStream	Initiate object could not be found.
410	RequiresInitiateStream2	Initiate object has been deleted.
400	RequiresInitiated	Part upload must specify the uuid or name of the initiated object.
400	RequiresLocalCluster	{0} value must refer to the local cluster on an administrative request.
412	RequiresLocalCluster2	{0} value must refer to the local cluster on an administrative request.
400	RequiresName	'Cid' queryArg requires a named content or domain specification.
403	RequiresOwned	Attempted owner access to a non-owned object.
403	RequiresRealm	Content specified unknown realm.
400	RequiresRecursiveQueryArg	All DELETES on a context must include the recursive query arg.
400	RequiresRemoteCluster	{0} value must refer to a remote cluster.

400	RequiresSourceHeader	{0} header not sent on request.
400	RequiresUUIDName	Missing or invalid UUID.
409	RequiresUniqueCreated	{0} must be unique.
409	RequiresUniqueVersion	{0} must be unique.
400	RequiresValidEncoding	Multipart writes must specify a valid EC encoding in the cluster, or on the request.
409	RequiresVersionHeader	{0} header requires {1} header.
409	RequiresVersionHeader2	{0} header requires {1} header.
0	SecurityRealmFailure	Failed to load context '{0}'.
0	SecurityStreamFailure	Failed to load object.
503	ServerFinalizing	Server cannot process client requests at this time: try again.
503	ServerFinalizing2	Server cannot process client requests at this time: try again.
503	ServerFinalizing3	Server cannot process client requests at this time: try again.
503	ServerInitializing	Server cannot process client requests at this time: try again.
503	ServerInitializing2	Server cannot process client requests at this time: try again.
503	ServerInitializing3	Server cannot process client requests at this time: try again.
0	UnexpectedException	Unexpected exception.
409	UnexpectedExceptionMeta	Unable to load object metadata for APPEND.
401	VariantAuthError	Unauthorized administrative request.
400	VariantAuthError2	The variant query argument may not be used on internal administrative requests.
400	VariantContextError	The variant query argument may not be used for context updates.
400	VariantContextError2	The variant query argument may not be used for context updates.
400	VariantForceDomainError	The forcedomain query argument may only be used on an unnamed variant COPY request.
412	VariantForceDomainError2	The forcedomain query argument cannot change an existing domain.
412	VariantForceDomainError3	The forcedomain query argument does not specify an existing domain.
400	VariantMethodError	The variant query argument may only be used with COPY or PUT methods, not {0}.
409	VersionCidInvalid	Object context mismatch. Domain specified for untenanted object.
409	VersionCidMismatch	Object context mismatch. Context id on request does not match context in object.
503	VersionDoesNotExist	Out-of-date version: try again.
503	VersionDoesNotExist2	Out-of-date version: try again.

404	VersionNotFound	Requested object was found with a delete marker.
401	VersionUnauthorized	Unauthorized administrative request.
500	VersionUnexpectedError	Error opening existing anchor object.
409	VersionUpdateError	Alias updates occurring too quickly. Make sure clocks are synced.
500	WriteFailure	{0}
500	WriteFailure2	{0}
500	WriteFailure3	{0}
500	WriteFailure4	{0}
409	WriterBundleDuplicate	Object already exists.
500	WriterBundleError	IOError creating bundled object.
500	WriterBundleException	Exception creating bundled disk object.
503	WriterBundleNoSpace	Not enough space in node.
410	WriterChecksumFailure	Checksum failure trying to GEN ec segment.
409	WriterDeleteNoExist	Object has been deleted.
400	WriterExcessMetaData	Too much persisted metadata.
412	WriterHeaderMismatch	{0} header does not match {1} header.
400	WriterInvalidContentLength	Content length for EC object exceeds maximum supported size ec.maxSupported.
409	WriterInvalidContentMD5	Calculated composite MD5 does not match provided composite MD5.
412	WriterInvalidEncoding	Invalid encoding header.
400	WriterInvalidExtent	'extentsize' must be greater than or equal to contentLength.
0	WriterInvalidExtentArg	'extentsize' must be an integer.
0	WriterInvalidInfo	Failed initiating info of part.
400	WriterInvalidManifest	Could not parse part manifest.
400	WriterInvalidPartDict	Invalid completion manifest. Parts entries must be dictionaries.
400	WriterInvalidPartDict2	Invalid completion manifest. Ranges entries must be dictionaries.
400	WriterInvalidPartId	Each part in the part manifest must contain a part number and a uuid, or a range.
400	WriterInvalidPartId2	Each part in the part manifest must contain a part number and a uuid.
400	WriterInvalidPartNumber	Part number found in part ({0}) does not match part number in completion manifest ({1}).
400	WriterInvalidPartUuid	Part uuid string was not valid UUID.

400	WriterInvalidPartUuid2	Part uuid string was not valid UUID.
400	WriterInvalidPath	x-castor-copy-source was not valid UUID.
400	WriterInvalidRange	x-castor-copy-source-range was not valid.
416	WriterInvalidRange2	x-castor-copy-source-range was not satisfiable. Cannot start past the end of the content.
416	WriterInvalidRange3	x-castor-copy-source-range was not satisfiable. Start must be less than end.
416	WriterInvalidRange4	x-castor-copy-source-range was not satisfiable. End extends past the end of the content.
400	WriterInvalidReplicationPost	Content-Type: {0} is only appropriate on an authorized admin POST.
412	WriterInvalidSegment	Segment is not part of this set.
400	WriterInvalidSetCount	Total EC encoded sets will exceed maximum number of sets for this segment size. Try a larger segment size.
400	WriterInvalidSort	Allowed values for sortOrder are 'part' and 'natural.'
400	WriterInvalidType	Unknown manifest type in segment header.
400	WriterInvalidType2	Unknown manifest type in manifest header.
400	WriterInvalidUploadId	UploadID in part does not match UploadID provided in request.
409	WriterLocked	A request for the specified object is already in progress on this node.
409	WriterLocked2	Simultaneous internode writes of same bundle UUID.
409	WriterLocked3	A request for the specified alias object is already in progress on this node.
409	WriterLocked4	A request for the specified named object is already in progress on this node.
400	WriterMD5Mismatch	Persisted {0} did not match value on request.
400	WriterMD5Mismatch2	Persisted {0} did not match value on EC request.
400	WriterManifestBadCO	contentOffset must be a number zero or greater.
400	WriterManifestBadSize	Size must be a number zero or greater.
500	WriterManifestIncomplete	Could not parse the manifest.
500	WriterManifestIncomplete2	Could not parse the manifest.
400	WriterManifestInvalidContentLength	Query arg 'contentLength' must be a number; at least zero.
400	WriterManifestInvalidPartOffset	Part offset must be a number of zero or greater.
400	WriterManifestInvalidPartUuid2	Could not form uuid from uuid string.
400	WriterManifestInvalidStartSize	Can not specify a size and offset that exceeds the bounds of a part, for part number {0}.

400	WriterManifestInvalidValue	Generated value must be a single character.
400	WriterManifestMissingPartNumber	Every uuid entry must have a part number.
400	WriterManifestMissingShrink	Truncating an object with PATCH requires the 'shrink' query arg.
400	WriterManifestNoContentRange	Every range must have a contentOffset specified.
400	WriterManifestNoSize	Every range must have a size specified.
400	WriterManifestOverlap	Content ranges can not overlap.
400	WriterManifestOverwrite	Can not write past specified contentLength.
400	WriterManifestUUIDAndValue	A range can not specify both a uuid and a generated value.
500	WriterMissingLocalMD5	A replica did not compute the Content-MD5.
500	WriterMissingRemoteMD5	A remote replica did not compute the Content-MD5.
409	WriterNotPermitted	Operation not permitted.
409	WriterNotPermitted2	Operation not permitted.
409	WriterOutOfDate	Cannot POST out-of-date version.
503	WriterOutOfMemory	Out of memory: try again.
503	WriterOutOfSpace	Not enough space in node.
409	WriterParallelBasisFail	Basis object is no longer ECed.
409	WriterParallelBasisFail2	Basis object is no longer ECed.
500	WriterParallelDeleteException	Unexpected exception trying to delete init manifest.
0	WriterParallelDeleteFail	Failed to delete init manifest.
503	WriterParallelObsolete	Object version found is obsolete; try again.
503	WriterParallelObsolete2	Object version found is obsolete; try again.
503	WriterParallelOpenFail	Failed to open object; try again.
503	WriterParallelOpenFail2	Failed to open object; try again.
0	WriterPartReadFailure	Unable to read part.
0	WriterProxyFailure	Replication peer request failed.
412	WriterProxyRequiresHeaders	Replication peer request failed.
503	WriterRequiresConnection	Unable to connect to peer.
409	WriterRequiresContentMD5	Missing required Content-MD5 on part {0}.
412	WriterRequiresCreated	{0} header is required.

0	WriterRequiresDestination	Replication peer failed to specify destination volume.
412	WriterRequiresEncoding	{0} header is required.
0	WriterRequiresExtentsize	'extentSize' argument is required for chunked transfer encoding.
412	WriterRequiresExtentsize2	'extentsize' query arg is required.
500	WriterRequiresExtentsize3	ProxyWriter requires extentSize when writing in chunked mode.
412	WriterRequiresGeneration	{0} header is required.
400	WriterRequiresMD5	Validating the Content-MD5 on an EC COPY requires an existing Content-MD5 stored on the object.
400	WriterRequiresManifest	The complete manifest was not provided.
500	WriterRequiresManifest2	Unable to create manifest.
500	WriterRequiresManifest3	Unable to create manifest.
500	WriterRequiresManifest4	Unable to create manifest.
500	WriterRequiresManifest5	Unable to create manifest.
500	WriterRequiresManifest6	Unable to create manifest.
500	WriterRequiresManifest7	Unable to write manifest.
412	WriterRequiresNodes	Unable to find sufficient nodes for replication.
507	WriterRequiresNodes2	Based on ec.protectionLevel=node, and ec.allowMultipleSegmentsPerLevel, this write cannot succeed.
500	WriterRequiresPartManifest	Invalid part manifest.
400	WriterRequiresPartlist	Completion manifest contained no valid parts.
400	WriterRequiresParts2	Part manifest must contain a populated 'parts' or 'ranges' key, but not both.
400	WriterRequiresPatch	Multipart upload must be initiated by PATCH to supply a range on contentLength on the completion.
400	WriterRequiresRange	Multipart upload by PATCH requires a 'range' key, and does not accept a 'parts' key on completion.
410	WriterRequiresRead	Unable to read part.
410	WriterRequiresRead2	Unable to read part.
503	WriterRequiresReplicas	Not enough replicas to succeed. Failing pipeline.
507	WriterRequiresReplicas2	Could not find at least two nodes to do replicate on write.
503	WriterRequiresReplicas3	No replicas were created.
412	WriterRequiresSegment	{0} header is required.

412	WriterRequiresSiblings	{0} header is required.
412	WriterRequiresSource	Could not find source object.
410	WriterRequiresStreams	Unable to read sufficient objects to complete GEN request.
507	WriterRequiresSubclusters	Based on ec.protectionLevel=subcluster, and ec.allowMultipleSegmentsPerLevel, this write cannot succeed.
412	WriterRequiresUniqueCreated	Unique {0} header is required.
400	WriterRequiresUniqueParts	Duplicate part number received in completion manifest.
503	WriterRequiresVolume	Targeted volume is not present.
503	WriterRequiresVolume2	Targeted volume is not available.
400	WriterSegmentOverflow	Request exceeds the segment limit.
400	WriterSegmentOverflow2	Request exceeds the segment limit.
503	WriterSequenceObsolete	Object version found is obsolete; try again.
503	WriterSequenceObsolete2	Object version found is obsolete; try again.
503	WriterShutdown	Node shutdown: try again.
400	WriterSizeLimit	EC encoded sets size exceeds supported size ec.maxSupported on APPEND.
412	WriterTimeout	Timed out waiting for peer write.
400	WriterTooMuchData	Too much data written for body.
500	WriterTooMuchData2	Object failed to generate digest.
400	WriterTypeConflict	An object may not be both a segment and a manifest.
401	WriterUnauthorized	Unauthorized internode request.
401	WriterUnauthorized2	Unauthorized internode request.
401	WriterUnauthorized3	Unauthorized internode request.
500	WriterUnexpectedException2	Unexpected exception in StreamWrite.
503	WriterVolumeFailed	Volume failed, try again.

SCSP Query Arguments

- [Query Arguments by Area](#)
 - [Note](#)
- [Support and Administration Arguments](#)
 - [Warning](#)

Put a question mark (?) after the URI, add the query argument, and put an ampersand (&) before any and all subsequent arguments to use arguments. See [RFC 3986, section 3.4](#).

```
http://URI?arg1=value&arg2=value&arg3=value
```

See [Search Query Arguments](#), which are specific to Swarm's Elasticsearch integration.

Query Arguments by Area

The following table describes all SCSP query arguments, which are grouped by feature or purpose.

- Query argument names are *case-insensitive*, as are most values.
- A valueless query argument (with no =) is changed internally to true.
- *Write* requests include [POST](#), [PUT](#), [APPEND](#), and [COPY](#).
- *Read* requests include [GET](#) and [INFO](#).
- Boolean values have equivalent forms: `alias`, `alias=yes`, `alias=true`

Applies to	Name	Value(s)	Description and usage
Alias Objects	alias	<code>yes/true</code> to activate	On write requests, indicates an alias object is created. The alias argument must be used with a POST method on an alias object and can optionally be included for other operations on alias objects.
Content Integrity	gencontentmd5	<code>yes/true</code> to activate	Computes the Content-MD5 for the body data of the request, returning the Content-MD5 as a header in the 201 Created response. Replaces the <code>Expect: Content-MD5</code> header, which is deprecated. (v9.2)
	hash	<code>hash value</code>	A content integrity hash value provided on the request for validation. The case of this argument does not matter.
	hashtype newhashtype	{ md5 sha1 sha256 sha384 sha512 }	Specifies the value of an object's hash. The <code>hashtype</code> query argument may appear on a variety of requests to generate or validate a content hash value. The <code>newhashtype</code> query argument is used to "re-seal" the content hash with one hash value while simultaneously checking another.
	validate	<code>yes/true</code> to activate	Validate On Read (VOR). Reads an object with an integrity seal. See Content Seals and Validation . On GET, validates data read from disk has not been corrupted. Swarm closes the connection before all data is sent if this check fails.
Domains	createdomain	value ignored	[Deprecated: v9.2] Add to a WRITE to create the domain specified by <code>domain=domain-name</code> . See Manually Creating and Renaming Domains .

	domain	domain-name	<p>Represents the domain name in some SCSP requests. Client applications most often send the domain name as the Host in the request. The domain argument can be supplied by the client to explicitly override any value from the Host request header when the Host header does not match the domain name. A domain argument always has precedence over the Host header in the HTTP/1.1 request.</p> <p>The sole situation domain is required is for an SCSP method on a domain object itself. Neither domain nor Host are required for requests <i>within</i> the default cluster domain.</p>
Erasure Coding (EC)	encoding	k : p	The integer values for the data (k) and parity (p) segment counts when specifying erasure coding.
	erasurecoded	yes/true to override, no /false to inhibit	<p>Used on EC writes to override the cluster's <code>policy.ecMinStreamSize</code> value.</p> <ul style="list-style-type: none"> • YES forces EC encoding for objects smaller than the cluster's minimum. • NO prevents EC encoding on objects that may otherwise be erasure-coded. <p>Using <code>erasurecoded</code> without <code>encoding</code> or when the cluster is not configured for EC results in a 400 Bad Request.</p>
	segmentsize	integer	The maximum size (in bytes) of a segment in any erasure-encoded set for this object, overriding the <code>ec.SegmentSize</code> configuration setting. This value cannot be smaller than 100 MB.
	segmentwidth	integer	Number of bytes. Allows <code>ec.segmentWidth</code> configuration value to be modified per request.
Listing Consistency	index	yes/true to activate, no /false to inhibit	<p>Appears on Gateway requests when enabling the Gateway Configuration option <code>EnhancedListingConsistency</code>. (v9.3)</p> <p>Optionally supplied on a POST, PUT, COPY, APPEND, or DELETE request. Performs synchronous search indexing of the newly written/deleted object.</p>
	sync	now or wait to activate, no /false to inhibit	<p>Appears on Gateway requests when enabling the Gateway Configuration option <code>EnhancedListingConsistency</code>. (v9.3)</p> <p>Optionally used on a listing query GET request to force consistency of results returned on the listing.</p> <ul style="list-style-type: none"> • <code>now</code> performs a refresh on the index in Elasticsearch immediately before performing the listing query. • <code>wait</code> delays execution of the listing query to provide Elasticsearch time to refresh the index.
Metadata	preserve	yes/true to activate	Works with COPY, PUT, and APPEND requests to verify custom metadata existing on the object is carried over on the write (see Custom Metadata Headers). Include the header name with the new value on the request to overwrite an existing value. Cannot be used with <code>replace</code> . (v9.2, v9.5)
	replace	yes/true to activate	Works with APPEND requests to remove any custom metadata existing on the object on the write, overriding the default APPEND behavior to preserve them (see Custom Metadata Headers). Include the header name with the new value on the request to add new metadata. Cannot be used with <code>preserve</code> . (v9.5)

Named Objects	newname	{new name for object, bucket, domain}	<p>Provides a new name (within the same bucket) for an update request (PUT, COPY, APPEND) on a named object. Requests for the original name return a 404 Not Found and the prior search metadata is removed after renaming an object.</p> <p>'Subdirectory' names are part of the object name, so they must be included as part of a <code>newname</code> query argument. The bucket name stops at the first slash. Everything after the first slash, including other slashes, is part of the object name.</p> <p><code>newname</code> also allows renaming domains and buckets. See Renaming Domains and Buckets.</p>
	putcreate	yes/true to activate, no/false to inhibit	<p>Allows use of HTTP PUT Create to create new named objects if set to yes. There is no need to add the <code>putcreate</code> query argument if the <code>scsp.allowPutCreate</code> storage setting is enabled.</p> <p>Directs Swarm to treat the request as a regular PUT if set to no, generating a 404 Not Found error if the named object does not exist.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>Although domains and buckets are named, Swarm applies all PUT requests on these objects as updates, regardless of the setting. Swarm fails the request with a 400 Bad Request error if <code>putcreate=yes</code> is used on a domain or bucket.</p> </div> <p>See SCSP WRITE.</p>
Multipart Write	inprogress	yes/true to activate	On a multipart PATCH complete, postpones HP segment consolidation until the object is completed again without the query argument, or by another method, such as COPY. (v9.4)
	shrink		On a multipart PATCH, required if the patch reduces the size of the object. (v9.4)
	partnumber	integer	On a multipart POST, indicates the part number of a multipart upload in progress. (v7.0)
	uploadid	upload id	On a multipart POST or DELETE, identifies all requests associated with a single multipart upload. It returns the ID as a 98-byte string. (v7.0)
	uploads	yes/true to activate	On a multipart POST, PUT, or APPEND, indicates the request is a multipart upload write initiate. (v7.0)
Recursive Delete	recursive	<p>yes/true to flag for deletion</p> <p>now for immediate reclamation</p>	<p><i>Required on DELETE of a bucket or domain (context).</i> Indicates when the health process may begin asynchronously reclaiming any content contained in the deleted context.</p> <p>This request creates a grace period based on the <code>health.recursiveDeleteDelay</code> Storage setting (the default is one week) unless using <code>recursive=now</code>. This grace period allows recreation of a bucket or domain before the content is lost. The health processor begins deleting all content contained in the deleted domain or bucket after the grace period ends. A critical error is logged indicating the object can be neither deleted nor accessed without the parent context if the deleted domain or bucket contains indelible objects. Added in v7.0.</p>
Replicate on Write (ROW)	count	integer	Used to affect the <code>replicate=immediate</code> behavior.

	replicate	<i>immediate, full, integer</i>	<p>Controls the response behavior to a POST, PUT, COPY, or APPEND request for a replicated object.</p> <ul style="list-style-type: none"> <i>immediate</i> - Two replicas must be created before Swarm sends the response; if more replicas are required, those additional replicas are created after the response. <i>full</i> - All required replicas must be created before Swarm sends the response. <i>integer</i> - Specifies the number of replicas to be created. As with <i>immediate</i>, two replicas must be created before Swarm sends the response. Swarm uses the smaller of those values if <code>policy.replicas max</code> or the lifepoint reps is less than this integer. <p>The above rules apply with <code>scsp.maxContextReplicas</code> taking the place of <code>policy.replicas max</code> if the object in the request is a bucket or a domain.</p>
SEND	feedid feedtype	all <i>integer</i> all search replication s3backup	<p>Specifies one or more specific feeds (<code>feedid=1&feedid=3</code>) as the replication destination. Open the Swarm UI, select Cluster > Feeds, and locate the ID column to find existing feed IDs.</p> <p>Specifies one or more types of feeds as the replication destination, from among these values: search, replication, s3backup.</p> <p>Use the special value "all" to refer to all feed IDs or types, including no feed.</p> <p>The SEND request needs query arguments for feedid, feedtype, or both, which is a union of all provided arguments; if neither are provided, SEND reverts to the legacy behavior.</p>
	timeout	true <i>number of seconds</i> false	<p>Sets how long to wait for replication to complete; if disabled (false; not recommended), feed processing can go on indefinitely if a feed is blocked.</p> <p>Using <code>timeout=true</code> waits for the Swarm setting <code>scsp.defaultFeedSendTimeout</code> time in seconds, which defaults to 30. Specifying a positive number for the timeout overrides the value in the Swarm setting.</p>
Versioning	version	ETag of desired object version	<p>Used on GET, HEAD, DELETE, COPY, APPEND, or SEND requests to specify a previous version to target on the request: <code>version={etag}</code></p> <p>Operations referencing the current version proceed normally if used in contexts where versioning is disabled, but any other ETag results in a 404 - Not Found. (v9.2)</p> <p>The query succeeds if the query argument value is the UUID (and ETag) of the object if used on GET or HEAD requests of an immutable object (which cannot be versioned). (v9.5)</p>
	suspendversioning		<p>Allows temporary suspension of version creation on POST, PUT, COPY, APPEND, and DELETE requests for versioned objects. It has the effect of updating the current version without adding to the versioning chain. (v9.5)</p>

Support and Administration Arguments

Name	Value(s)	Description and usage
admin	<i>yes/true</i> to activate	<i>Use with requests by the 'admin' user.</i> An override used to rescue content from being stranded with no ability to delete or update it due to an overly restrictive Allow header.
aliasuuid	domain-UUID	<p>Used on an administrative override COPY request to specify the alias UUID (CID) of an existing domain or bucket for the purpose of changing the name. It must be used with the <code>newname</code> argument, which provides the new domain or bucket name. This is used to resolve name collisions that may occur during replication.</p> <p>See Resolving Duplicate Domain Names.</p>

checkintegrity	yes/true to activate	<p>Used on a GET of an erasure-coded object to get a summary of what segments exist in the manifest and the locations in the cluster. The body of the request has this information instead of the object.</p> <p><code>checkintegrity</code> is a fast and efficient way to check the integrity of an erasure-coded object, for verifying all segments exist before executing a GET.</p> <p>Swarm responds with 200 (OK) if it finds enough segments to recreate the object; else it returns 412 (Precondition Failed) and error text in the body of the response lists the missing segments.</p>
cid	UUID	<p>A Context Identifier (CID) can be used to access an otherwise inaccessible object. See Accessing Inaccessible Objects with CID.</p>
countreps	yes/true to activate or diskless	<p>Used to have the HEAD return the number and location of online replicas in the cluster. The count returned is for the object manifest for an erasure-coded object.</p>
examine	yes/true to activate	<p>On a GET or HEAD request, triggers an immediate health processor examination of the request object rather than waiting for the health processor to revisit the object as part of the normal cycle.</p>
ignoreerrors	yes/true to activate	<p>Used on an erasure-coded GET request to step over any broken EC sets and generate any data it can. Use of this query argument activates chunked transfer encoding.</p>
recreatecid	domain or bucket alias UUID	<p>Used for certain administrative actions, such as a special POST request to recreate a particular domain or bucket with the specified alias UUID. This is often performed after deleting a domain or bucket that orphans contents. See Restoring Domains and Buckets.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p> Warning</p> <p>Do not attempt to use <code>recreatecid</code> as a method to move bucket contents across domains within the cluster.</p> </div>
redir	yes/true to activate, no/false to inhibit	<p>Asks the request PAN to encourage or inhibit redirection on the request.</p>
verbose	yes/true to activate	<p>Used on a GET or HEAD request to have all relevant headers returned in the response. (v8.1)</p> <p>Includes a <code>MinReps</code> header that reports the number of replicas that exist (applies to wholly replicated objects and segments).</p>

HTTP Response Codes

Following are HTTP response codes that may be received from Swarm, with notes about any Swarm-specific meaning.

See [RFC 7231 section 6](#) of the HTTP/1.1 specifications for information about response status codes.

Response	Methods	Notes
100 Continue	POST, PUT, APPEND	See SCSP WRITE , WRITE for Large Files (Expect: 100-continue). See SCSP Compatibility and Support , Issues with 100-Continue Header
101 Switching Protocols	-	Informs the client about the server switching the protocols to the one specified in the Upgrade message header field during the current connection.
200 OK	GET, HEAD, DELETE	Standard response for successful requests. For EC, indicates Swarm found enough segments to recreate the object, which is a prerequisite for success.
201 Created	POST, PUT, COPY, APPEND	The success response for a POST or PUT request.
202 Accepted	POST	Request accepted (such as for a multipart completion), but not processed.
203 Non-Authoritative Information	-	Returned meta information was not the definitive set from the origin server.
204 No Content	-	Request succeeded without requiring the return of an entity-body.
205 Reset Content	-	Request succeeded but requires resetting of the document view that caused the request.
206 Partial Content	GET	Successful response to a GET that includes one or more Range headers, returning the specific range data.
207 Multi-Status	-	
300 Multiple Choices	-	Requested resource has multiple choices at different locations.

<p>301</p> <p>Moved Permanently</p>	<p>All</p>	<p>Resource permanently moved to a different URL</p> <p>Requests the client to resend the current request to the location supplied in the response headers <i>and</i> to direct all future requests to that new node until further notice. The <code>location</code> header supplies the authorization parameter to be included in the request to the new PAN.</p>
<p>302</p> <p>Found</p>	<p>-</p>	<p>Requested resource was found under a different URL but the client should continue to use the original URL.</p>
<p>303</p> <p>See Other</p>	<p>-</p>	<p>Requested response is at a different URL and can be accessed only through a GET command.</p>
<p>304 Not Modified</p>	<p>GET, HEAD, PUT, APPEND, COPY</p>	<p>If-Modified-Since condition not met. Requested object was not modified since the last request.</p> <p>Requested object was not modified since the time specified in the If-Modified-Since header. The response is returned without any message-body.</p> <p>If-None-Match condition not met on GET or HEAD.</p>
<p>305</p> <p>Use Proxy</p>	<p>-</p>	<p>Requested resource should be accessed through the proxy specified in the location field.</p>
<p>307</p> <p>Temporary Redirect</p>	<p>All</p>	<p>Resource is temporarily moved to a different URL.</p> <p>The client should resend the current request to the <i>location</i> supplied in the response headers, but to continue using the original PAN for the next request until further notice.</p> <p>See Application Best Practices, <i>Use Workarounds for Redirects</i>.</p>
<p>400</p> <p>Bad Request</p>	<p>All</p>	
<p>401 Unauthorized</p>	<p>All</p>	<p>Occurs as the normal <i>initial</i> response of HTTP authentication for a domain.</p> <p>Administrative request lacks an Authorization header with suitable administrative credentials. The response includes a WWW-Authenticate challenge containing the administrative domain named Castor administrator and other required items.</p>
<p>403</p> <p>Forbidden</p>	<p>Varies</p>	<p>Unsupported method was used.</p> <p>Alias objects only support POST, DELETE, GET, and HEAD.</p> <p>Domain objects only support GET, HEAD, COPY, PUT, and APPEND.</p>

<p>404 Not Found</p>	<p>GET, HEAD, APPEND, PUT, COPY, DELETE</p>	<p>Indicates that the content cannot be located in this cluster. Common causes include these:</p> <ul style="list-style-type: none"> Object deletion Request errors (such as the wrong bucket name) Network failure Node(s) down for maintenance Timeouts due to a heavily loaded or extremely active cluster Rebalancing due to new drive capacity in the cluster Requesting aliased objects without using the etag flag, because the overlay index does not keep primary UUIDs for aliased objects. Appending to an immutable object or a UUID that does not exist. Multipart range index out of range. Attempting to read a version not associated with this alias.
<p>405 Method Not Allowed</p>	<p>-</p>	<p>Method specified in the Request-Line was not allowed for the specified resource.</p> <p>APPEND not allowed for content-encoded objects.</p> <p>Allow header forbids this method on this object.</p>
<p>406 Not Acceptable</p>	<p>-</p>	<p>Resource requested generates response entities that has content characteristics not specified in the accept headers.</p> <p>Content-encoding not acceptable, or not found on request or in 'decoderSettings' setting.</p>
<p>407 Proxy Authentication Required</p>	<p>-</p>	<p>Request requires the authentication with the proxy.</p>
<p>408 Request Timeout</p>	<p>-</p>	<p>Client fails to send a request in the time allowed by the server.</p>

<p>409 Conflict</p>	<p>PUT, POST, APPEND, COPY, DELETE</p>	<p>Request was unsuccessful due to a conflict in the state of the resource. Includes the following conditions:</p> <ul style="list-style-type: none"> Attempting to create a domain or bucket object which already exists. Renaming a domain or bucket to a name which already exists. Attempting to update an immutable object. Attempting to update an alias object in a domain that does not exist. Rapid updates of an object, resulting in the error "Later version already exists." Persisted Content-MD5 did not match value on request. Basis object is no longer erasure-coded. Encoding has changed on the basis object since the initiate. Initialized object must have matching uploadID, which might be updated since the initiate.
<p>410 Gone</p>	<p>GET, HEAD, DELETE</p>	<p>Delete marker.</p> <p>Erasure-coded object: too few segments found to service request, checksum failure attempting to generate an EC segment, or unable to read sufficient objects to complete request.</p>
<p>411 Length Required</p>	<p>-</p>	<p>Server cannot accept the request without a valid Content-Length header field.</p> <p>Content-Length must be provided for all non-EC object requests.</p> <p>Content-Length must be provided and must be zero for COPY request.</p>

<p>412 Precondition Failed</p>	<p>All</p>	<p>Precondition specified in the Request-Header field returns false.</p> <p>For erasure coding, indicates Swarm cannot recreate the object, listing the missing segments in the body of the response.</p> <p>For named objects, indicates that the named object already exists or that the bucket or domain cannot be found.</p> <p>For unnamed objects, indicates Swarm did not write the object because the domain does not exist, if cluster.enforceTenancy is set to true.</p> <p>See WRITE for Unnamed Objects.</p> <p>For replication, indicates that the cluster cannot locate at least two nodes to initially store the replicas. Check the Replica-Count header to verify Swarm creates the correct number of replicas.</p> <p>When Swarm initiates a replication request to a PAN and the replication or initial write fails, Swarm fails <i>both</i> procedures and generates a 412. This guarantees that two copies of the object are saved on separate nodes in the cluster before returning a 201 response.</p>
<p>413 Request Entity Too Large</p>	<p>-</p>	<p>Server is not ready to receive the large file and has closed the connection.</p> <p>See WRITE for Large Files (Expect: 100-continue).</p>
<p>414 Request-URI Too Long</p>	<p>-</p>	<p>Request unsuccessful because the URL specified is longer than the server can process.</p>
<p>415 Unsupported Media Type</p>	<p>-</p>	<p>Request unsuccessful because the entity of the request is in a format not supported by the requested resource.</p>
<p>416 Requested Range not satisfiable</p>	<p>GET</p>	<p>GET request includes invalid Range headers because the request is out of bounds of the data.</p> <p>Invalid range header, cannot parse.</p> <p>x-castor-copy-source-range was not satisfiable. Cannot start past the end of the content, and start must be less than the end.</p>

<p>417 Expectation Failed</p>	<p>-</p>	<p>Expectation given in the Expect request-header was not fulfilled by the server.</p> <p>gencontentmd5 query argument or Expect: Content-MD5 header (deprecated) is not supported for APPEND.</p> <p>gencontentmd5 query argument or Expect: Content-MD5 header (deprecated) is not allowed on multipart write initiate request.</p> <p>List operations do not support any 'Expect' headers.</p>
<p>500 Internal Server Error</p>	<p>All</p>	<p>Critical error in Swarm. Check logs for more information and contact DataCore Support if necessary.</p> <p>Tip – A 500 error is almost always accompanied by a non-500 'child' error with more detail, which is returned in the error headers. See Error Response Headers.</p>
<p>501 Not Implemented (Forbidden Feature)</p>	<p>-</p>	<p>Requested method is unsupported in Swarm. Only the methods listed in the Allow header currently work in Swarm.</p> <p>Erasure coding is not enabled in the configuration.</p> <p>List operations unavailable because Elasticsearch is not configured or licensed.</p> <p>List operations require a Search Feed, but none are currently available.</p> <p>Destination server version is invalid or does not support remote replication.</p> <p>Destination server is not recognized, or destination server name is not provided.</p> <p>Destination <i>cluster.name</i> is not set, or differs from expected.</p>
<p>502 Bad Gateway</p>	<p>-</p>	<p>Server received an invalid response from the upstream server while attempting to fulfill the request.</p> <p>Cannot find a source for RETRIEVE operation.</p> <p>Castor-System-Cluster header has an invalid cluster name.</p>
<p>503 Service Unavailable (Try Again)</p>	<p>All</p>	<p>Processing was interrupted, so the client should attempt again. Common causes include the following:</p> <ul style="list-style-type: none"> • Swarm cannot complete due to a transient problem or inadequate resources to process the request. • Cluster is too busy to service additional requests. • Gateway cannot communicate with Swarm.

504 Gateway Timeout	-	Upstream server failed to send a request in the time allowed by the server. Cannot connect to destination.
505 HTTP Version not supported	All	Indicates a request was received with an HTTP version other than HTTP/1.1. Swarm only supports HTTP/1.1.
507 Insufficient Storage Space	PUT, POST, APPEND, COPY	Request cannot be completed because of space limitations or licensing restrictions/errors.

Search Queries

Swarm Storage integrates with Elasticsearch to allow client applications to list and search metadata on objects being stored in the Swarm cluster. Access these operations by applying query arguments are specific to search.

See [Elasticsearch for Swarm](#) for details on implementing and managing Elasticsearch.

Scope of Searching in Swarm

Swarm Search supports domain-level searching; to search on an entire cluster, iterate across the domains. Swarm looks up objects by the underlying `contextid` so querying can be performed by context (domain/bucket) name as usual but always get correct query results even if a domain or bucket has been renamed. Swarm generates the final content query after this lookup, which supports a wide range of functionality:

- **Filter by name or value.** Filter by an object's `name` or any metadata (`field`) using equality checks, greater/less than comparisons, and wildcard matches.
- **Filter by buckets.** Filter the search to certain buckets (`context`) using greater/less than comparison (for buckets in numbered ranges) or wildcard matches (for buckets that match a prefix pattern). Include the bucket in the URL path to restrict search to a single bucket.
- **Filter by object type.** Add the `stype` argument to filter by one of these Swarm types: *domain*, *bucket*, *named*, *alias*, *immutable*, *unnamed* (both alias and immutable), or *all*.
- **Operate on metadata fields.** Perform AND and OR operations on the values of metadata fields to find the matching objects.
- **Sort by value.** Sort specification can combine multiple metadata fields, including the context.
- **Paginate large result sets** with sort markers. Apply markers when the context is sorted (ascending or descending) and in conjunction with markers for other metadata fields, when the sort specification includes multiple fields.
- **Calculate disk usage.** The `du` aggregation filters the results to calculate disk usage, inclusive or exclusive of object replicas in the cluster.
- **Locate versions.** Use the `versions` argument to surface all historical versions of a single object or of all objects in the context if using [versioning](#).

- [Metadata Field Matching](#)
- [Listing Operations](#)
- [Search Operations](#)
- [Search Examples](#)
- [Walkthrough: Ordering sets of filtered objects](#)
- [Search Query Arguments](#)

Metadata Field Matching

- [Matching examples](#)
 - [Return JPEG images](#)
 - [Return JPEG or PNG images](#)
 - [Return JPEG or PNG images in a bucket](#)
 - [Match a single positional wildcard](#)
- [Searchable metadata fields](#)
 - [Tip](#)
 - [Basic metadata fields](#)
 - [Full metadata fields](#)
 - [Hyphen conversion](#)

In addition to specifying query arguments, metadata field matching criteria are specified in the URI. This allows for fine-tuning the result set to return objects in the storage cluster matching one or more matching criteria.

Matching criteria are logically AND expressions by default and can be switched to OR expressions using the **or=yes** query argument. The context for the search, everything after the domain name, and any value for the prefix argument are always considered to be logical AND constraints for the match. A bucket name or prefix pattern must match even when **or=yes** is used if are specified in the URI.

Wildcards can be used to match field values:

- ? – wildcard for a single character
- * – wildcard for multiple characters

The following examples show the different matching concepts.

Matching examples

❶ CLUSTER or <cluster> in a URL stands for <host>[:<port>], where *host* is a fully qualified domain name or IP address, plus a *port* number if other than 80. If the Host header does not match the domain name, override it with the `domain=` argument.

Return JPEG images

```
GET http://{cluster}/
?format=json
&content-type=image/jpeg
```

Return JPEG or PNG images

```
GET http://{cluster}/
?format=json
&or=yes
&content-type=image/jpeg
&content-type=image/png
```

Using glob-style pattern matching:

```
GET http://{cluster}/
?format=json
&content-type=image/*
```

Return JPEG or PNG images in a bucket

{png OR jpeg} AND "pics" bucket:

```
GET http://{cluster}/pics
?format=json
&or=yes
&content-type=image/jpeg
&content-type=image/png
```

Match a single positional wildcard

This example finds values including "grey" or "gray":

```
GET http://{cluster}/
?format=json
&x-color-meta=gr?y
```

Searchable metadata fields

Each Search Feed indexes metadata for searching, but *which* metadata depends on how [the feed is defined](#) in the Storage UI (or legacy Admin Console). With the **Search full metadata** checkbox selected, Swarm indexes *all* available metadata for the objects in the cluster; with it unselected, Swarm indexes basic metadata fields to support listing operations.

Allow for [additional storage and RAM](#) on the search servers to support it if **Search full metadata** is implemented on the search feed.



Tip

Searching on the basic metadata fields can be performed even without **Search full metadata** enabled.

Basic metadata fields

The following table provides a list of the standard, baseline field names as they are mapped between the name used in the query argument values and the name given in the XML and JSON output formats. Notice that the output name may be different from the name used in the query argument and that the output name can change depending upon the output format.

Query Argument	XML Name	JSON Name	Description
tmBorn	LastModified	last_modified	Time of create or last update. The query argument may use either UTC date-time or Unix timestamp (float) in search requests. The microseconds and time-of-day portions of a UTC date-time are both optional.
content-length	Size	bytes	Size in bytes
name	Key	name	UUID or name using URL encoding
content-type	content-type	content_type	Content type
etag	ETag	hash	Entity tag
sizewithreps	sizewithreps	sizewithreps	Number of bytes using the maximum reps value

Full metadata fields

The following list shows the metadata field names that are indexed for full metadata search. [Custom Metadata Headers](#) are included in these patterns.

- castor-* (except castor-system-*)
- content-base
- content-disposition
- content-encoding
- content-language
- content-location
- content-md5
- lifepoint
- x-*-meta[-*]

These fields do not show up in listings unless *explicitly* included using the [fields query argument](#), like this:

```
GET http://{cluster}/mybucket
?format=xml
&fields=name,content-length,x-color-meta
```

See [Search Operations](#).

Case: While the metadata field names are case-insensitive for the purposes of matching, they are stored in the cluster as given during the WRITE operation. The metadata field *values* are case-sensitive.



Hyphen conversion

Custom metadata field names that contain hyphens (-) have these characters converted to underscores (_) in the result output. Swarm allows variants with either hyphens or underscores on input, but it favors underscores on output.

For [multipart uploads](#), both `Castor-System-Uploadid` and `Castor-System-Partnumber` allow query argument2 to use either hyphens or underscores in the field name, as is supported for `content-type`. (v10.2)

Listing Operations

- [Encoding](#)
- [Listing domain contents](#)
 - [Basic form of listing a domain](#)
 - [Filter buckets that start with a string](#)
 - [Listing unnamed objects within a domain](#)
 - [Listing everything within a domain](#)
- [Note](#)
- [Listing a bucket context](#)
 - [Listing bucket contents](#)
- [Note](#)
- [Listing untenanted unnamed objects](#)
 - [Listing untenanted unnamed objects](#)
- [Storage in use](#)
 - [Important](#)
- [Domain storage in use](#)
 - [Space used by all content in domain](#)
 - [Space used by named objects](#)
 - [Space used by unnamed objects](#)
- [Bucket storage in use](#)
 - [Calculating space used by named objects in a bucket](#)
- [Annotations in existence](#)
 - [Listing annotation objects for given ETag](#)

Listing operations are a specialized class of searching that usually have a context constraint of a domain or a bucket, except in the case of listing untenanted unnamed objects or finding annotation objects. When performing listing operations, the user is typically interested in the hierarchy or membership within a context. Since listing operations are in fact searches, you can use other searching options and metadata constraints in combination with them.

Encoding

When non-ASCII characters are included, list query response bodies are UTF-8 encoded. When reading and writing non-ASCII characters, applications *must* decode the response body from UTF-8 prior to interpreting the list body.

Listing domain contents

Domains are a context that contains bucket objects and unnamed objects. Buckets are identified by name. Unnamed objects are identified by UUID and are either mutable (alias) or immutable.

Basic form of listing a domain

```
GET /?format=json&domain=myDomain
```

Filter buckets that start with a string

```
GET /?format=json&domain=myDomain&prefix=Southwest_
```

Listing unnamed objects within a domain

```
GET /?format=json&domain=myDomain&stype=unnamed
```

Listing everything within a domain

```
GET /?format=json&domain=myDomain&stype=all
```



Note

Swarm assumes **stype=bucket** when listing a domain, so it returns a list of bucket names unless you request a specific **stype**.

Listing a bucket context

Buckets are a specific context type that belongs to a domain and can contain only named objects. This operation is similar to listing a domain context with additional URI element of the bucket name added.

Listing bucket contents

```
GET /myBucket?format=json&domain=myDomain
```



Note

Swarm assumes **stype=named** when listing a bucket because no other stype is valid.

Listing untenanted unnamed objects

If the storage cluster includes unnamed objects that are not contained in a domain (untenanted), you may still list those objects by specifying an empty domain context.

Listing untenanted unnamed objects

```
GET /?format=json&domain=&stype=unnamed
```

Storage in use

You can dynamically query the storage in use for a domain or bucket context using the **du** query argument. This is valid for domains and buckets.

- **du=withreps** requests the total storage impact of objects.
- **size=0** prevents the return of the request body (the calculated value is returned in the header).

Argument	Header Result	Description
----------	---------------	-------------

du=withoutreps	CastorBytesUsed: 2189243	For du=withoutreps , the HTTP/1.1 response header returns a Castor-System-Bytes-Used field that indicates a summary of the storage used by the objects within that context. Swarm calculates the space from the sum of the body bytes of all relevant and distinct objects. This calculation does not consider the number of replicas for each distinct object. .
du=withreps	CastorBytesUsedWithReps: 109416625	For du=withreps , the HTTP/1.1 response header returns a Castor-System-Bytes-Used-With-Reps field that indicates a summary of the storage used by the objects within that context. Swarm calculates the space from the sum of the body bytes of all relevant objects using each object's maximum reps value from the object lifepoint headers and the assigned value from the cluster multiplied by the object's size. A 100 MB object with reps=2 consumes 200 MB of space. The same object with reps=3 consumes 300 MB of space. Stored with an erasure coding value of reps=4:2, the object consumes 150 MB of storage.

If the object has three lifepoints that include all previous example values, the maximum of reps=3 is chosen for the calculation and the storage impact is recorded as 300 MB. Every object has the metadata field **sizewithreps** that records its space impact.



Important

The **Castor-System-Bytes-Used** and **Castor-System-Bytes-Used-With-Reps** fields are computed on-demand. Depending upon the object count within the context, they can consume computational resources on the search servers. Applications should only request **du** operations when the space calculations are required.

Domain storage in use

When querying the storage in use for a domain, you have the option of selecting the types of objects to consider using the **stype** argument.

Argument summary:

- **du=withreps** requests the full storage impact of objects.
- **size=0** prevents the return of the request body (the calculated value is returned in the header).
- **stype=named|unnamed|all** selects the types of objects included in the calculation.

Space used by all content in domain

```
GET /?format=json&domain=myDomain&du=withreps&size=0&stype=all
```

```
HTTP/1.1 200 OK
Gateway-Request-Id: 26F809F67D883E6D
Content-Type: application/json; charset=utf-8
Castor-System-Object-Count: 17
Castor-System-Bytes-Used-With-Reps: 121590
[snip]
```

Space used by named objects

```
GET /?format=json&domain=myDomain&du=withreps&size=0&stype=named

HTTP/1.1 200 OK
Gateway-Request-Id: C6A8D293950C1FD5
Content-Type: application/json; charset=utf-8
Castor-System-Object-Count: 12
Castor-System-Bytes-Used-With-Reps: 121422
[snip]
```

Space used by unnamed objects

```
GET /?format=json&domain=myDomain&du=withreps&size=0&stype=unnamed

HTTP/1.1 200 OK
Gateway-Request-Id: 3D79D93B73A07E35
Content-Type: application/json; charset=utf-8
Castor-System-Object-Count: 2
Castor-System-Bytes-Used-With-Reps: 168
[snip]
```

Bucket storage in use

When querying the storage in use for a bucket, there are only named objects within the bucket context. The **stype** argument is not required.

Argument summary:

- **du=withreps** requests the total storage impact of objects.
- **size=0** prevents the return of the request body (the calculated value is returned in the header).

Calculating space used by named objects in a bucket

```
GET /mybucket?format=json&domain=myDomain&du=withreps&size=0

HTTP/1.1 200 OK
Gateway-Request-Id: 100601F9E31D5ECC
Content-Type: application/json; charset=utf-8
Castor-System-Object-Count: 1000
Castor-System-Bytes-Used-With-Reps: 22016
[snip]
```

Annotations in existence

To retrieve any annotation objects that may exist in the cluster for a given object, submit a listing query that sets the argument "decorates" equal to the ETag of the target object in question:

Listing annotation objects for given ETag

```
GET /?format=json&domain=&decorates=8c2c582c216a1f088c3652bced5a5f91
```

See [Metadata Annotation](#).

Search Operations

- [Best practice](#)
- [Performance Impact](#)
 - [Tip](#)
- [Search Examples](#)
 - [Searching within a domain](#)
 - [Searching within a bucket](#)
 - [Searching by multiple field matching](#)
 - [Using fields= to return specific field names](#)
- [Enabling case-insensitive search \(name.lower\)](#)
 - [Important](#)
- [Using content-length](#)
 - [Tip](#)
 - [Using content-length](#)
 - [Important](#)

Search operations are an extremely powerful feature for locating content; they work on the metadata of the objects within the storage cluster. Searches use metadata matching constraints provided in the client request and return a list of objects that match those constraints.

Searches can take place across all objects within the domain or searches can be constrained to the context of a particular bucket. When full metadata search is enabled, you can use *any* custom metadata field value as a search constraint.

Best practice

Do not apply a context filter redundantly in cases where Swarm filters by default, such as when searching for named objects in a bucket, buckets in a domain, or unnamed objects in a domain.

Performance Impact

Consider performance when designing searches spanning entire domains. `context` (domain and bucket) names are looked up from the `contextid`, so these domain-wide searches incur an additional performance penalty:

- Retrieving the `context` field
- Sorting on the `context` field
- Filtering on the `context` field

Tip

Check the performance impact when sorting and filtering across an entire domain.

Search Examples

Unless otherwise noted all matching operations are string-based comparisons:

Searching within a domain

```
GET /?format=json&domain=myDomain
&content-type=application/pdf
```

Searching within a bucket

```
GET /myBucket?format=json&domain=myDomain
&content-type=application/pdf
```

Searching by multiple field matching

```
GET /?format=json&domain=myDomain
&x-color-meta=red
&content-type=image/png
```

Using fields= to return specific field names

```
GET /myBucket?format=json&domain=myDomain
&fields=name,content-length
```

Enabling case-insensitive search (name.lower)

You control case-sensitivity in your Elasticsearch queries by using the correct form of the **name** field:

- **name** field: ES searches are case-sensitive, so searching `FOO` matches *only* `FOO`
- **name.lower** field: ES searches are case-insensitive (as if all values are lowercase), so searching `FOO` matches `FOO`, `Foo`, `foo`

The Swarm search setting, [search.caseInsensitive](#), is specific to SCSP queries, versus querying ES directly. When this is enabled, case-insensitive SCSP search queries are performed by default. (v9.0)

Swarm Setting	Effect
<code>search.caseInsensitive = 1</code>	All name-based searches use the name.lower field, so SCSP named searches are always case-insensitive.
<code>search.caseInsensitive = 0</code>	All name-based searches use the name field, and is case-sensitive.

Important

Custom metadata values are always indexed to be only case-sensitive or case-insensitive, depending on the value of `search.caseInsensitive`. If an index is built with the wrong setting, you must change the setting and build a new index.

Using content-length

The **content-length** field for objects is recognized as a numeric field and supports equality, less-than-equal-to, and greater-than-equal-to matching operators.



Tip

Use ">=" or ">" and "<=" or "<".

Using content-length

```
GET /?format=json&domain=myDomain&content-length=1024
GET /?format=json&domain=myDomain&content-length<=1024
GET /?format=json&domain=myDomain&content-length>=1024
```

**Important**

The `Content-MD5` metadata field cannot be used as a search constraint, either alone or with other fields. Use it only in the output fields for a search.

Search Examples

- [Queries for Buckets](#)
 - [Tip](#)
 - [Query with fields, not involving context](#)
 - [Query with fields, retrieving context as one of the fields](#)
 - [Retrieving context as one of the fields and sorting on context](#)
 - [List objects >1GB, sorted by size descending](#)
 - [Sorting on multiple fields, context, and name, in different orderings](#)
 - [Listing query, retrieving context as one of the fields, sorting on multiple fields with context](#)
- [Queries for Named Objects](#)
 - [Retrieving context as one of the fields](#)
 - [Retrieving context as one of the fields and sorting on context](#)
 - [Sorting on context and another field in different orderings](#)
 - [Sorting and inequality context filters](#)
- [Queries for Unnamed Objects](#)
 - [With context sort](#)
- [Queries for Named and Unnamed Objects](#)
 - [With context sort](#)

These examples of how to search Swarm are demonstrated through curl.

i CLUSTER or <cluster> in a URL stands for <host>[:<port>], where host is a fully qualified domain name or IP address, plus a port number if other than 80. If the Host header does not match the domain name, override it with the domain= argument.

Queries for Buckets

i **Tip**

For queries within a single bucket, include the bucket in the URL; the filter context=<domain>/<bucketname> does not need to be added.

Query with fields, not involving context

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length"
```

Query with fields, retrieving context as one of the fields

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context"
```

Retrieving context as one of the fields and sorting on context

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&sort=context:asc"
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,contextid,context&sort=context:desc"
```

List objects >1GB, sorted by size descending

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&content-length>=1073741824&sort=content-length:desc,name&fields=name,content-length,context"
```

Sorting on multiple fields, context, and name, in different orderings

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&sort=context:asc,name:asc"
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&sort=context:desc,name:desc"
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&sort=name:asc,context:asc"
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&sort=name:desc,context:desc"
```

Listing query, retrieving context as one of the fields, sorting on multiple fields with context

```
curl -i -X GET
"http://{cluster}/bucket1?format=json&domain=example.com
&fields=name,content-length,context&sort=content-length:desc,context:desc"
```

Queries for Named Objects

Retrieving context as one of the fields

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED"
```

Retrieving context as one of the fields and sorting on context

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=context:asc"
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=context:desc"
```

Sorting on context and another field in different orderings

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=context:asc,content-length:asc"
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=context:desc,content-length:desc"
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=content-length:asc,context:asc"
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=content-length:desc,context:desc"
```

Sorting and inequality context filters

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=content-length:desc,context:desc
&context>=example.com/2015"
```

Sorting and wildcard context filter

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=content-length:desc,context:desc
&context=example.com/us-*"
```

Sorting and context marker and multiple markers

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,siz,context&stype=NAMED&sort=context:asc
&marker=example.com/bucket2"
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=content-length:asc,context:asc
&marker=15,example.com/bucket1"
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=context:asc,content-length:asc
&marker=example.com/bucket1,15"
```

With du argument, withreps and withoutreps

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=content-length:desc,context:desc
&du=withoutreps"
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,content-length,context&stype=NAMED&sort=content-length:desc,context:desc
&du=withreps"
```

Queries for Unnamed Objects

With context sort

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,tmborn,context&stype=IMMUTABLE&sort=context:asc"
```

Queries for Named and Unnamed Objects

With context sort

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,tmborn,context&stype=IMMUTABLE,NAMED&sort=context:asc
&du=withreps"
```

With context wildcard filter

```
curl -i -X GET
"http://{cluster}?format=json&domain=example.com
&fields=name,tmborn,context&stype=IMMUTABLE,NAMED&sort=context:asc
&context=example.com/foo*&du=withoutreps"
```

Walkthrough: Ordering sets of filtered objects

The following are details and guidance for a complex example, how to paginate (list ordered subsets of) the search results on objects matching specific metadata.

This walkthrough shows how and why to combine use of three related [Search Query Arguments](#): `size`, `marker`, and `sort`.

- [How to count objects in a bucket](#)
- [How to count filtered objects](#)
- [How to limit \(page\) the results](#)
- [How to pull the next result set](#)
- [How to use the hash as marker](#)

How to count objects in a bucket

This query returns an empty set (`size=0`), focus on the header output alone:

```
$ curl -si -u jdoe "https://jdoe.cloud.acme.com/public/
?format=json&domain=jdoe.cloud.acme.com&size=0"
Enter host password for user 'jdoe':
HTTP/1.1 200 OK
Date: Wed, 16 Dec 2020 15:55:42 GMT
Gateway-Request-Id: 5BF093C3AECC45AD
Server: CASTor Cluster/12.0.0
Via: 1.1 jdoe.cloud.acme.com (Cloud Gateway SCSP/7.1.0)
Gateway-Protocol: scsp
Allow-Encoding: *;q=0
Castor-System-Alias: ac611714399ae0e5f22a628d4e8c26f4
Castor-System-CID: 924273bee8a6e01865d7b2a315ea5ae3
Castor-System-Cluster: foo.tx.acme.com
Castor-System-Created: Thu, 10 Sep 2015 19:45:24 GMT
Castor-System-Name: public
Castor-System-Version: 1441914324.106
X-Last-Modified-By-Meta: jdoe@
X-Owner-Meta: jdoe
X-Timestamp: Thu, 10 Sep 2015 19:45:24 GMT
X-timestamp: Wed, 16 Dec 2020 15:55:42 GMT
Content-Type: application/json;charset=utf-8
Castor-Object-Count: 62
Castor-System-Object-Count: 62
Last-Modified: Wed, 16 Dec 2020 15:55:42 GMT
Transfer-Encoding: chunked

[
]
```

Check the value for `Castor-Object-Count` to determine how many objects are associated with the search performed. The number of objects in the "public" bucket under domain "jdoe.cloud.acme.com" is 62 per above.

How to count filtered objects

Drill down further and focus on items matching a metadata characteristic. Filter for a specific kind of content (application, audio, image, text, video) being stored in the object, which is recorded in the `Content-Type` metadata header. Note: filter objects by custom metadata as well.

This search filters for objects holding MP4 video content:

```
$ curl -si -u jdoe "https://jdoe.cloud.acme.com/public/
?format=json&domain=jdoe.cloud.acme.com&size=0&content-type=video/mp4"
Enter host password for user 'jdoe':
HTTP/1.1 200 OK
Date: Wed, 16 Dec 2020 17:10:58 GMT
Gateway-Request-Id: C28EB97FE6EF3914
Server: CASTor Cluster/12.0.0
Via: 1.1 jdoe.cloud.acme.com (Cloud Gateway SCSP/7.1.0)
Gateway-Protocol: scsp
Allow-Encoding: *;q=0
Castor-System-Alias: ac611714399ae0e5f22a628d4e8c26f4
Castor-System-CID: 924273bee8a6e01865d7b2a315ea5ae3
Castor-System-Cluster: foo.tx.acme.com
Castor-System-Created: Thu, 10 Sep 2015 19:45:24 GMT
Castor-System-Name: public
Castor-System-Version: 1441914324.106
X-Last-Modified-By-Meta: jdoe@
X-Owner-Meta: jdoe
X-Timestamp: Thu, 10 Sep 2015 19:45:24 GMT
X-timestamp: Wed, 16 Dec 2020 17:10:58 GMT
Content-Type: application/json;charset=utf-8
Castor-Object-Count: 39
Castor-System-Object-Count: 39
Last-Modified: Wed, 16 Dec 2020 17:10:58 GMT
Transfer-Encoding: chunked

[

]
```

Filtering the "public" bucket in domain "jdoe.cloud.acme.com" for MP4 content (`content-type=video/mp4`) produces a count of 39 videos (`Castor-Object-Count: 39`).

How to limit (page) the results

Limit the *size* of the search results when a portion of the search results is needed or the entire set of objects is too large to be displayed in full. Combining three search query arguments provides the control needed:

- `size` – Controls the size of the result set, unrelated to object size (`content-length`). Set it to 0 when the actual listing is not needed.
- `marker` – Used with `size` to paginate large result sets. Use an empty key to begin a new search, then use the last `sort` key value of the results on the next request to continue pagination.
- `sort` – Sorts the results on one or more fields, in the order listed. Sorting defaults to ascending, so add descending (`:desc`) as needed. Sorting is computationally intensive, so sort output when necessary.

```

$ curl -s -u jdoe "https://jdoe.cloud.acme.com/public/
?format=json&domain=jdoe.cloud.acme.com&content-type=video/mp4&marker=&size=5&sort=etag:desc"
Enter host password for user 'jdoe':
[
  {
    "last_modified": "2018-09-04T17:14:44.848000Z",
    "bytes": 261671693,
    "name": "recording-a.mp4",
    "hash": "ff3ea60737felaec9b4a506a23c29fe9",
    "written": "2018-09-04T17:14:44.848000Z",
    "accessed": "2018-09-04T17:14:44.848000Z",
    "content_type": "video/mp4"
  },
  {
    "last_modified": "2017-07-31T15:37:45.580000Z",
    "bytes": 77337274,
    "name": "recording-b.mp4",
    "hash": "f2402263315cad55c0909f50f7154c13",
    "written": "2017-07-31T15:37:45.580000Z",
    "accessed": "2017-07-31T15:37:45.580000Z",
    "content_type": "video/mp4"
  },
  {
    "last_modified": "2017-06-14T18:32:28.592000Z",
    "bytes": 24926795,
    "name": "recording-c.mp4",
    "hash": "ed35d20e43af0a5a1757f000905ff653",
    "written": "2017-06-14T18:32:28.592000Z",
    "accessed": "2017-06-14T18:32:28.592000Z",
    "content_type": "video/mp4"
  },
  {
    "last_modified": "2019-07-19T15:50:53.444000Z",
    "bytes": 3810394,
    "name": "recording-d.mp4",
    "hash": "ec3c93febe2ff19e3c6a6561f8c25363",
    "written": "2019-07-19T15:50:53.444000Z",
    "accessed": "2019-07-19T15:50:53.444000Z",
    "content_type": "video/mp4"
  },
  {
    "last_modified": "2018-06-29T19:02:45.724000Z",
    "bytes": 55816215,
    "name": "recording-e.mp4",
    "hash": "e7e4a3d4cd8ee0df2894520d0624ceca",
    "written": "2018-06-29T19:02:45.724000Z",
    "accessed": "2018-06-29T19:02:45.724000Z",
    "content_type": "video/mp4"
  }
]
    
```

- Skip getting the return headers of the request: this is performed because total object account above for objects filtering for is already determined.
- "marker=", set to empty, starts at the beginning of the result set.
- "size=5" returns the first 5 of our filtered objects.
- "sort=etag:desc" sorts the objects in descending order from the "hash" value (ETag) associated with the object which is covered in detail below.

How to pull the next result set

Choose a *marker* for the subsequent query to get the next five results in the set. Subsequent requests can be selected (marked) by a characteristic (metadata field) returned for the last object in the set. There are many to choose from:

```
{
  "last_modified": "2018-06-29T19:02:45.724000Z",
  "bytes": 55816215,
  "name": "recording-e.mp4",
  "hash": "e7e4a3d4cd8ee0df2894520d0624ceca",
  "written": "2018-06-29T19:02:45.724000Z",
  "accessed": "2018-06-29T19:02:45.724000Z",
  "content_type": "video/mp4"
}
```

The best practice is to use the "hash" field:

Field	Downsides of use as a marker
name	Effort: Must URL-encode any special characters Not guaranteed to be unique <i>except</i> inside a given bucket
last_modified	Not guaranteed to be unique Can introduce gaps in paging the result sets Changeable in real time, during the query run itself
hash	None

The "hash" is the object's ETag (entity tag), which is guaranteed to be unique across the entire cluster. It supports queries spanning multiple buckets and domains.

How to use the hash as marker

It takes two steps to page through result sets using the hash value as the marker:

1. Parse the hash value out of the output for the *last* object in the previous set.
2. Set the marker argument to be the hash string.

The hash value listed for the last object in the result above is "e7e4a3d4cd8ee0df2894520d0624ceca", so start our next search for results after the object as follows:

```

$ curl -s -u jdoe "https://jdoe.cloud.acme.com/public/
?format=json&domain=jdoe.cloud.acme.com&content-type=video/mp4&marker=e7e4a3d4cd8ee0df2894520d06
Enter host password for user 'jdoe':
[
  {
    "last_modified": "2017-09-01T16:10:01.496000Z",
    "bytes": 23902924,
    "name": "recording-f.mp4",
    "hash": "e7d46a777a2c67f5ebc016f8a8626ac5",
    "written": "2017-09-01T16:10:01.496000Z",
    "accessed": "2017-09-01T16:10:01.496000Z",
    "content_type": "video/mp4"
  },
  {
    "last_modified": "2018-05-10T20:05:19.500000Z",
    "bytes": 57463240,
    "name": "recording-g.mp4",
    "hash": "df9fc440b0a230e2d771e29b08829fe8",
    "written": "2018-05-10T20:05:19.500000Z",
    "accessed": "2018-05-10T20:05:19.500000Z",
    "content_type": "video/mp4"
  },
  {
    "last_modified": "2016-01-04T21:42:05.896000Z",
    "bytes": 76657180,
    "name": "recording-h.mp4",
    "hash": "cf65f6fac3d9683be29b6e37f1bc5910",
    "written": "2016-01-04T21:42:05.896000Z",
    "accessed": "2016-01-04T21:42:05.896000Z",
    "content_type": "video/mp4"
  },
  {
    "last_modified": "2017-02-28T19:45:25.664000Z",
    "bytes": 312294768,
    "name": "recording-i.mp4",
    "hash": "b89823900fcbe09c762f9946cf598612",
    "written": "2017-02-28T19:45:25.664000Z",
    "accessed": "2017-02-28T19:45:25.664000Z",
    "content_type": "video/mp4"
  },
  {
    "last_modified": "2017-09-08T21:26:44.148000Z",
    "bytes": 442920798,
    "name": "recording-j.mp4",
    "hash": "b6e556acd26d43f052490afd0fe42e4f",
    "written": "2017-09-08T21:26:44.148000Z",
    "accessed": "2017-09-08T21:26:44.148000Z",
    "content_type": "video/mp4"
  }
]
    
```

This returns the next set of 5 objects in descending ETag value ordering (`sort=etag:desc`).

For the next set, parse out the hash for the last object listed (`b6e556acd26d43f052490afd0fe42e4f`) and continue until walking through all objects returned.



Important

The "sort" argument is computationally intensive. Watch the load on the Elasticsearch cluster to gauge the performance impact when running queries like this.

Search Query Arguments

- [marker Argument](#)
- [stype Argument](#)

Argument	Values	Usage	Example
format	json xml	<p>Required. Requests a search query and specifies the desired output format. This query argument is required to trigger a domain/bucket listing query.</p> <p>This format applies to the body portion of the HTTP/1.1 response and does not affect the format of the response headers. All search operations are available with either format.</p>	&format=json &format=xml
context	string	<p>Filters the search to certain buckets one of these ways:</p> <ul style="list-style-type: none"> • Greater/less than comparison, for buckets in numbered ranges: 0123, 0124, 0125, ... (context>=0124) • Wildcard matches, for buckets matching a prefix: us-tx, us-ky, us-tn, us-ga, ... (context=us-*) <ul style="list-style-type: none"> • Wildcards are supported for matching on prefixes, so use an asterisk at the <i>end</i> of the string. • Wildcard matching across many buckets may impact performance. <p>To restrict search to a bucket, include the bucket in the URL path. Always use with use domain={domain-name}.</p>	&context<=example.com/2015 &context=example.com/foo*
contextid	UUID	Specifies a search filtered to the context identified by the UUID given.	&contextid=80a4957...
decorates	UUID	Use in a listing query to find any annotation objects that may exist for a given ETag (or earlier query result "hash"). See Metadata Annotation and Listing Operations .	&decorates=a95780a4...
domain	domain-name	<p>In a normal SCSP storage request, the required Host header is the assumed domain (or context) for the operation. Most HTTP/1.1 clients and browsers use the host portion of the URL as the value for the Host header in the Swarm request. Supply the domain argument to explicitly override it if the value of the Host request header does not match the domain name. A domain argument always has precedence over the Host header in the HTTP/1.1 request.</p> <p>Set the value equal to nothing (empty string) to find unnamed objects not tenanted in any domain.</p>	&domain=example.com &domain=
domains	none (ignored)	Used for listing the domains within a cluster. No value is used with this argument.	&domains

<p>du</p>	<p>withoutreps yes true withreps</p>	<p>Dynamically queries the disk usage for a domain or bucket context. With du, use the <code>size=0</code> query argument (to shorten the output by preventing the return of the request body) and add any filters needed. The computed result value appears in the <code>Castor-System-Bytes-Used[-With-Reps]</code> response header.</p> <ul style="list-style-type: none"> <code>withoutreps</code> sizes the <i>content uploaded</i>: it finds the sum of the bytes of the unique objects being stored. <code>withreps</code> sizes the <i>storage impact</i>: it finds the underlying disk space impact of the objects stored in a domain or bucket context, a value weighted by the expected number of replicas in the cluster, which is an approximation of the data footprint of the results. <p>See Storage in Use for how to query the storage in use for domains and buckets.</p>	<p>&du=withreps&size=0</p>
<p>field-name</p>	<p>any indexed metadata field, such as: content-length</p>	<p>Filters results to those matching the operations on one or more specified fields. Multiple filters are joined by a logical AND; to specify OR, add the <code>&or</code> argument.</p> <p>Supports standard comparison operators on specific fields to filter the results returned:</p> <ul style="list-style-type: none"> <code>a = b</code>, equal to <code>a <= b</code>, less than or equal to <code>a >= b</code>, greater than or equal to <p>This is an overloading of normal query argument processing.</p>	<p>&x-width-meta<=800 &x-height-meta<=600 &content-type=image/bmp&or &content-type=image/png &content-length>=1073741824</p>
<p>fields</p>	<p>Comma-separated list of field names, or <code>all</code></p>	<p>Replaces the default fields and specifies, as a comma-separated list, which fields to display in the output (see Metadata Field Matching).</p> <p>Use <code>fields=all</code> on bucket and domain listing requests to return all available fields on each record in the response. (v9.2)</p>	<p>&fields=name,content-length &fields=all</p>
<p>index</p>	<p>yes/true to activate, no/false to inhibit</p>	<p>On a write (POST, PUT, COPY, APPEND, DELETE) request, forces immediate search indexing of the newly written/deleted object.</p> <p>Appears on Gateway requests when enabling the Gateway Configuration option <code>EnhancedListingConsistency</code>. (v9.3)</p>	<p>&index=yes</p>
<p>sync</p>	<p>now or wait to activate, no/false to inhibit</p>	<p>On a read (listing query GET) request, forces consistency of results returned on the listing.</p> <ul style="list-style-type: none"> <code>now</code> performs a refresh on the index in Elasticsearch immediately before performing the listing query. <code>wait</code> delays execution of the listing query to provide Elasticsearch time to refresh the index. <p>Appears on Gateway requests when enabling the Gateway Configuration option <code>EnhancedListingConsistency</code>. (v9.3)</p>	<p>&sync=now</p>
<p>marker</p>	<p>empty value or a comma-separated list of sorted column key values</p>	<p>Provides a mechanism to use multiple requests to receive a complete result set. Defaults to <code>&sort=name</code>.</p> <p>Used with the <code>size</code> argument to paginate large result sets. Use an empty key to begin a new search. Use the last sort key value of the results on the next request to continue pagination from that place on the next request.</p>	<p>&marker=&sort=tmBorn&size=50</p>
<p>or</p>	<p>yes/true to activate</p>	<p>Joins them with a logical OR (default is AND) when used with multiple field comparisons. Defaults to false.</p>	<p>&or</p>

prefix	string	<p>Matches the string to the start (prefix) of the object's name or UUID. The names of the objects in the output list all share the same prefix string.</p> <p>Named objects are always contained within a bucket, and the first "/" following the bucket name is the delimiter. Any additional slashes following the first slash are part of the object's name. That is, for <code>/photo/Q1/apple.jpg</code>, where <code>photo</code> is the bucket, retrieve the object using <code>prefix=Q1</code>.</p> <p>Unnamed objects require the additional argument <code>stype=unnamed</code> to match on UUIDs.</p> <p>To find untenanted unnamed objects, include <code>"domain="</code> (empty string).</p>	<p>&prefix=Q1</p> <p>&stype=unnamed&prefix=93f</p> <p>&stype=unnamed&prefix=93f&domain=</p>
size	integer	<p>Constrains the number of results to be returned by the query. Defaults to 1000. This argument is equivalent in purpose to the "limit" or "max-keys" arguments in other cloud protocols.</p> <p>Important: This is the size of the result set, <i>not</i> the object size. To work with object size, use the field <code>content-length</code>.</p> <p>Tip: Set size to 0 when no actual listing is needed, such as when using the "du" query argument.</p>	&size=5000
sort	Comma-separated list of field names, with optional modifier <code>:asc :desc</code>	<p>Sets the sort order of one or more fields, with the first field sorting first. Sorting direction defaults to ascending (<code>:asc</code>); to specify descending (<code>:desc</code>), add the modifier to one or more of the fields.</p> <p>Important: Sorting exacts a computational penalty, so sort output only when necessary.</p>	&sort=size:desc,name
stype	Comma-separated list of Swarm types	<p>Specifies the type of objects being requested in a search. Defaults to <code>all</code>.</p> <p>Valid values: <code>domain, bucket, named, alias, immutable, unnamed, all</code></p>	&stype=unnamed
versioned	<code>true false</code>	<p>Allows filtering updateable (named and alias) objects by versioning status, checking whether versioning was enabled in the context at the time they are written. Checks the value of the <code>Castor-System-IsVersioned</code> header. (v10.0)</p>	&versioned=true
versions	<code>true (no value) previous</code>	<p><code>True</code> (or no value) requests a listing of all versions of all objects, including the current.</p> <p><code>Previous</code> requests past versions (v10.2).</p>	&versions
versions&name	name of object	Requests a listing of all versions of the named object.	&versions&name=logo.jpg
versions&prefix	first part of name	Requests a listing of all versions of all objects matching the prefix.	&versions&prefix=2019Q1/
		See Object Versioning .	

marker Argument

The `marker` argument provides a mechanism to retrieve a single result set using multiple HTTP requests. This is useful when a large result set is impractical to receive using one HTTP request. All marker operations work using a sorted result set. The system implicitly uses `sort=name` if a `sort` argument is not provided. The `size` argument is used to control how many items are returned per HTTP request.

On the first request, `marker=` (without a value) is used to indicate a request of the beginning of the result set. For subsequent requests, the last value(s) from the sort field(s) is used to indicate the last record received from the previous request.



Tip

Use a UTC date-time or Unix timestamp (float) when using `tmborn` as the marker. The microseconds and time-of-day portions of a UTC date-time are both optional.

Marker Example

Consider the following set of object names in the bucket `dictionary`.

```
applaud
appoints
arches
basically
boardwalk
buffers
carpet
defender
```

Receive the first four items in the result set:

```
GET /dictionary?format=json&domain=example.com&size=4&fields=name&marker=
[
  { "name": "applaud" },
  { "name": "appoints" },
  { "name": "arches" },
  { "name": "basically" }
]
```

The last sort field value from the first request is `basically`, so the next request to continue receiving the result set is as follows:

```
GET /dictionary?format=json&domain=example.com&size=4&fields=name&marker=basically
[
  { "name": "boardwalk" },
  { "name": "buffers" },
  { "name": "carpet" },
  { "name": "defender" }
]
```

Since the result set is now exhausted, a subsequent request yields:

```
GET /dictionary?format=json&domain=example.com&size=4&fields=name&marker=defender [ ]
```

Sort and Marker Relationship

The marker value is the *last* value or values from the sort fields. The marker contains as many last field values in it as well if using: `sort={field1},{field2},...,{fieldN}`:

```
marker={lastValueField1},{lastValueField2},...,{lastValueFieldN}
```

Both ascending and descending sort order is supported for each of the sort order fields.

Determining Completion

The set of objects in the cluster is continually changing so list/search requests are never complete. A criterion exists for considering marker requests complete. The `Castor-System-Object-Count` response header returns the number of objects remaining to enumerate including the records retrieved by the current request.

A client may consider the iterating request complete when the `CastorObject-Count` value equals the number of records retrieved by the current request.

Markers across the Domain

A strategy for using markers to retrieve a result set of a search across the entire domain is to sort based upon a unique tuple of metadata fields all objects have. An example:

```
sort=context,name
```

See [Baseline Metadata Fields](#).

The `name` field is a UUID or an application assigned string depending upon the object type.

stype Argument

The **stype** argument allows filtering search results to certain types of objects. The following values can be used:

stype	Description	Notes
all	(default) All object types	
bucket	Bucket objects	Must reside in a domain. <code>stype=bucket</code> is assumed implicitly when listing a domain, as this is a common operation.
named	Content objects with an application-supplied name	Must reside in a bucket. Search across buckets by specifying a domain and not including a bucket in the URL.
unnamed	Content objects with a Swarm-assigned UUID	Cannot reside in a bucket. Tenanted in a domain unless cluster.enforceTenancy is disabled.
immutable	Immutable unnamed objects	A subset of <code>stype=unnamed</code>
alias	Mutable unnamed objects	A subset of <code>stype=unnamed</code>

Avoid sending invalid requests when using **stype**, such as specifying **stype=unnamed** when searching within a bucket (which cannot contain unnamed objects). **stype=named** in a domain context is valid, even though named objects exist in a bucket, because the bucket is contained *within* the domain context as well.

Use the **unnamed**, **immutable**, or **alias** value of `stype` to select the desired set to list the unnamed objects in the domain. Include `domain=` set to nothing (empty string) to find untenanted unnamed objects. Untenanted unnamed objects return an empty string for the domain field in search and listing results.



Important

Unnamed objects are guaranteed to be tenanted in a domain when the **cluster.enforceTenancy** configuration option is enabled. Include `domain=` (empty string) to find untenanted objects.

Connecting to a Swarm Cluster

This section describes how your application can connect to a storage cluster or node.

Requests to store, retrieve, or delete objects in the cluster can initially be sent to any accessible node. The cluster decides which node is best suited to carry out the request, based on resource availability and other factors.

Primary access node (PAN)

The node that initially fielded the request is called the **PAN**.

Secondary access node (SAN)

If the PAN did not field the request, it sends the application a redirect request that includes the address of a node referred to as the **SAN**. Using this method, a storage cluster performs automatic and intrinsic load balancing.

Any node in a storage cluster can serve as a PAN. The PAN for a particular request is the first node in the cluster that receives the request from the application. After a PAN receives a request, it decides whether to service the request itself or to request the application to redirect it to one of the other nodes in the cluster.

Because the PAN assumes responsibility for having the application direct the request to another node in the cluster, your application does not need to provide any load balancing solution. Swarm implements load balancing, even when nodes are dynamically added or removed from the cluster.

- [Choosing How to Access a PAN](#)

Choosing How to Access a PAN

Because any node can be called on by the PAN to service any particular request, all nodes must be accessible to the application client. An application can locate a PAN to use for transactions using one of the following methods.

To locate a PAN, use one of these methods (listed most preferred to least):

- [Use the Swarm SDK](#)
- [Use Multicast-DNS \(mDNS\)](#)
 - [Important](#)
- [Use DNS round robin](#)
 - [Tip](#)
- [Use a pool of static IP addresses](#)
- [Use a single static IP address](#)

Use the Swarm SDK

(recommended) Integrate applications with Swarm using the Software Development Kit (SDK). Along with the convenience it provides, applications can use the **ProxyLocator** or **StaticLocator** object included with the SDK to locate and communicate with a node.

The SDK's **ProxyLocator** subclass performs two functions:

- Performs a GET / to the SCSP Proxy to pre-populate the local list of Swarm node IP addresses.
- Dynamically maintains this list as redirects and other responses are received directly from Swarm nodes.

See the [SDK Overview](#).

Use Multicast-DNS (mDNS)

Another way to make nodes locate an initial PAN is to use [mDNS](#). mDNS is often referred to as *Zeroconf*, the collective name for DNS and DNS Service Discovery to enable zero-configuration networking.

mDNS is supported for all deployments. It provides the most flexibility because it presents applications with a list of storage nodes to choose from when selecting a PAN without requiring the application to maintain a static list of IP addresses.

Every Swarm node implements an mDNS service that allows applications to provide [service discovery](#). Even if DHCP is used to assign and change node IP addresses, mDNS allows an application to "discover" an active node in any storage cluster and use it as the PAN. Several free mDNS client implementations in various languages are available online for implementing mDNS node location.



Important

Verify the `cluster.name` parameter value is unique for each cluster when using mDNS. The parameter is located in the `node.cfg` file or in the Platform Server's cluster configuration.

Swarm mDNS support allows an application to discover all nodes on a network, all nodes in a specific cluster, or to look up a node. To implement this process, it "publishes" several different records, including an A (host) record for the node and an SRV (service) record under the `_scsp._tcp` service type.

Although an in-depth description of mDNS deployment is beyond this scope, a typical use example is provided below. This example uses the [Avahi](#) command line tools to pass in the name of the cluster and return all nodes discovered in the cluster. Two nodes are found and the IP addresses are returned in the `address` field for each record.

```
% avahi-browse -tr
_clustername._sub._scsp._tcp local + eth0 IPv4 D2024267FF8F1DD056EEA15E40EE52C9
_scsp._tcp local = eth0 IPv4 CD35B28FD2E70CD1E47095C774F8050F
_scsp._tcp local hostname = [CD35B28FD2E70CD1E47095C774F8050F.local]
address = [192.168.1.123] port = [80] txt = [] = eth0 IPv4 D2024267FF8F1DD056EEA15E40EE52C9
_scsp._tcp local hostname = [D2024267FF8F1DD056EEA15E40EE52C9.local]
address = [192.168.1.125] port = [80] txt = []
```

Use DNS round robin

For large and/or dynamic storage clusters where nodes are often added and removed (even for temporary maintenance), address the cluster using a DNS host name instead of an IP address.

This method is recommended for all deployments. It is particularly helpful for multi-tenancy, as the DNS name can be used to pass in a domain.

The domains must resolve to at least one IP address (such as an "A" record) for client applications so the application software includes a recognized Swarm domain name in the Host header of the HTTP/1.1 request when using DNS with multi-tenancy.

 **Tip**

With some DNS servers, move the maintenance of the PAN addresses out of the applications and into the DNS server itself. The Berkeley Internet Name Domain ([BIND](#)), the most commonly used DNS server, allows entry of multiple "A" records that map a single DNS name to more than one IP address.

This process also requires static IP addresses, but it enables the application to use a single DNS name (or multiple DNS names if using multiple domains) for the entire cluster. The DNS server selects one of the defined IP addresses on a round-robin basis. The application must resolve the host name again if one of the nodes does not respond.

Use a pool of static IP addresses

A less desirable approach for an application to address a storage cluster is to use a stored list of several or all static IP addresses for the nodes in the cluster. This method is *not* recommended for a production environment.

The application's stored list of IP addresses must be accessible programmatically from the application. The application can attempt another IP address if one of the nodes fails to respond to a request.

The application is able to add the new IP address to the list if a redirect response reveals a storage node not in the original list. This may be a good approach if the cluster is relatively stable with respect to static node IP addresses. Do not use this method if nodes are frequently added and removed from the cluster.

Use a single static IP address

The simplest but least recommended (and least supported) way for an application to address a storage cluster is to assign a static IP address to at least one of the cluster nodes and then use that IP address in every request. It is recommended to restrict usage to a development environment. It can be set up quickly, but is not maintainable in a larger system.

The simplicity of this approach is balanced by a significant disadvantage. The application cannot send requests to the cluster if the sole PAN is taken out of service or fails for any reason, even though other nodes may still be functioning and all desired content is still available.

SCSP Methods

These are general restrictions on methods:

- **Immutables** – The only supported methods for unnamed immutable objects are GET, HEAD, POST, and DELETE; other methods return a 403 (Forbidden) response.
- **Domains** – The only supported methods for domains are GET, HEAD, COPY, PUT, and APPEND. POST and DELETE require cluster administrator credentials and special consideration.



Best practice

If you cannot use the [Content UI Overview](#) to create, edit, and delete your domains, use the [legacy Admin Console](#).

- [SCSP READ](#)
- [SCSP INFO](#)
- [SCSP WRITE](#)
- [SCSP DELETE](#)
- [SCSP UPDATE](#)
- [SCSP APPEND](#)
- [SCSP COPY](#)
- [SCSP SEND](#)

SCSP READ

- [READ for named objects](#)
- [READ for unnamed objects](#)
- [READ for erasure-coded objects](#)
- [READ with content validation](#)
- [READ for node status and cluster capacity](#)
- [READ with range headers](#)
 - [Note](#)
 - [Note](#)

This section provides general information about the SCSP READ method that applies to both named and unnamed objects.

READ is a request to the storage cluster for a specific object. The READ request is formatted as an HTTP request using the GET method.

SCSP Method	HTTP Method	RFC 7231 Section
READ	GET	4.3.1

READ for named objects

```
GET /mybucket/samplefile.txt HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
```

READ for unnamed objects

```
GET /12BFEA648C2697A56FD5618CAE15D5CA HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
```

Swarm makes no assumptions about User-Agent (except it is an HTTP/1.1 client). The Host header is mandatory and must conform to the requirements of [Section 14.23](#) in the HTTP/1.1 specification.

READ for erasure-coded objects

Erasur coding objects on disk allows storing large objects in the cluster with a smaller storage footprint, compared to earlier versions of Swarm.

See [Working with Large Objects](#).

READ is affected by the **checkIntegrity=yes** query argument used to verify all segments are found before executing the READ.

See [SCSP Query Arguments](#).

READ with content validation

To validate the content during a read, add the query argument **validate=yes** to the URI:

```
GET /name-or-uid?validate=yes HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
```

Using this argument, the content digest is computed and compared at the end. A hash mismatch causes the socket connection to be dropped before sending the final bytes.

READ for node status and cluster capacity

A READ request can be submitted without a name or UUID:

```
GET / HTTP/1.1
Host: cluster.example.com
User-Agent: CAStor Client/0.1
```

Using this argument returns counts for high-level node methods (READ, WRITE, DELETE, ...), as well as cluster-wide *Space Available* and *Total Capacity* values.

This information is also available on the Node Status page that appears when navigating to a node's IP address with the designated SCSP port (for example, <http://192.168.99.100:80>).

READ with range headers

In some cases, an application may be interested in a byte portion of a larger object stored in Swarm. Rather than read the entire object and filter out the interesting parts, the application can include one or more *Range headers* with an SCSP READ request. A READ request can include more than one Range header.

See the [HTTP/1.1 specification](#) for a thorough discussion of Range headers.

Below are some examples and interpretations:

Range	Returns
0-499	First 500 bytes of the object
500-999	The second 500 bytes
-500	Last 500 bytes
0-499, 500-999	First 1000 bytes



Note

Range headers are not compatible with either integrity seals or the ContentMD5 header because both require a hash of the object's entire contents. The connection is closed as if the [integrity seal](#) or the Content-MD5 was invalid if the range is not set to a value greater than or equal to the size of the object.

READ requests that include invalid Range headers (ranges not existing in the object) cause Swarm to respond with a 416 (Range not satisfiable) error. A successful response to a READ that includes one or more Range headers is 206 (Partial content). Data in the requested ranges are included in the 206 response.

READ requests that include a range such as Range: 0-199, 300-349, 500-999 returns a Content-Type: multipart/byteranges response consisting of three parts: 200, 50, and 500 bytes of content.

See [14.35 \(Range\)](#) and [Appendix 19.2 \(Internet Media Type multipart/byteranges\)](#) in [HTTP/1.1 RFC](#) .

**Note**

Entering a range with the range in reverse order (where the end of the range is entered first) returns the entire object. For example, Range: 999-500 returns all content in the object. The range header is essentially ignored.

- [Error Responses to READ](#)
- [Normal Responses to READ](#)

Error Responses to READ

The storage cluster can return various types of responses when the specified content cannot be found or there is a problem with the READ request.

- [400 Bad Request for READ](#)
- [404 Not Found for READ](#)
- [416 Requested range not satisfiable for READ](#)

400 Bad Request for READ

The response below indicates a problem with the READ request, such as missing mandatory headers, invalid message body, or any other violation of HTTP/1.1 by the GET request. The actual reason for the error is described in the message body of the response.

```
HTTP/1.1 400 Bad Request
Date: Wed, 1 Sept 2012 15:59:02 GMT
Server: CASTor Cluster/5.0.0
Connection: close
Content-Length: 24
Content-Type: text/html
CRLF Host header is required.
```

404 Not Found for READ

The following response indicate the requested content cannot be located in this cluster because of one of the following reasons:

- The object was deleted.
- The request included errors (for example, the wrong requested bucket name).
- Network failure.
- The node (all nodes containing the requested object) is down for maintenance.
- Timeouts occur due to a heavily loaded or extremely active cluster.

This response can occur after additional drive capacity is added to the cluster. The cluster attempts to rebalance the objects from heavily loaded nodes to less-loaded nodes when it recognizes the new storage drive.

It is recommended to retry the request after a 404 error message for applications recognizing a given object exists.

```
HTTP/1.1 404 Not Found
Content-Length: 97
Content-Type: text/html
Date: Tue, 14 Jun 2012 01:12:16 GMT
Server: CASTor Cluster/6.0.0
Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE
<html><body><h2>CASTor Error</h2><br>Bucket example.com/bucket failed to load (404)</body></html>
```

416 Requested range not satisfiable for READ

The response below indicates one or more range headers supplied in the request are out of bounds with respect to the data.

```
HTTP/1.1 416 Requested range not satisfiable
Date: Wed, 1 Sept 2012 15:59:02 GMT
Server: CASTor Cluster/5.0.0
Connection: close
Content-Length: 24
Content-Type: text/html
Range specs out of range
```

Normal Responses to READ

The storage cluster can return the following responses for the requested content:

- [READ response for domain](#)
- [READ response for bucket](#)
- [READ response for named object](#)
- [READ response for unnamed objects](#)
- [READ response for range headers](#)
- [READ response for moved permanently](#)
- [READ response for moved temporarily](#)

The content can be returned directly to the node that sent the request or redirected to another node in the cluster.

See [SCSP Headers](#) for a list of response headers.

READ response for domain

Example response to a READ or INFO request for an object in a domain.

The initial HTTP 401 Unauthorized response is a normal part of HTTP authentication for a domain. Because access to a domain requires domain manager credentials, you always see an initial HTTP 401 Unauthorized response for an INFO request in a domain.

```

HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest
  realm="cluster.example.com/_administrators",
  nonce="05d0a60eelf44361f449496505e05116",
  opaque="784d8bc3fe3a48a5105b4f8ddd8ae0e7",
  stale=false, qop="auth", algorithm=MD5
WWW-Authenticate: Basic
  realm="cluster.example.com/_administrators"
Content-Length: 51
Content-Type: text/html
Date: Sat, 16 Oct 2012 00:41:23 GMT
Server: CASTor Cluster 5.0.0
Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE
HTTP/1.1 200 OK
Castor-Authorization: cluster.example.com/_administrators, POST=
Castor-System-Alias: ec87e3c7c410cc04fc4c838061898d9c
Castor-System-CID: ffffffffffffffffffffffffffffffff
Castor-System-Cluster: cluster.example.com
Castor-System-Created: Fri, 15 Oct 2012 23:59:40 GMT
Castor-System-Name: cluster.example.com
Castor-System-Owner: admin@CASTor administrator
Castor-System-Version: 1287187180.959
Content-Length: 0
Last-Modified: Fri, 15 Oct 2012 18:35:56
GMT lifepoint: [] reps=16
Etag: "da8bfbb04d089b9c22ae77747f327233"
Date: Sat, 16 Oct 2012 00:41:23 GMT
Server: CASTor Cluster/5.0.0

```

READ response for bucket

Example response to a READ request for a bucket:

```
HTTP/1.1 200 OK
Castor-System-Alias: d36dfca69ba7752f4708b1fa9bf9918b
Castor-System-CID: ec87e3c7c410cc04fc4c838061898d9c
Castor-System-Cluster: cluster.example.com
Castor-System-Created: Fri, 15 Oct 2012 18:36:05 GMT
Castor-System-Name: bucket
Age: 62
Castor-System-Version: 1287167765.255
Content-Length: 0
Content-Type: application/x-www-form-urlencoded
Last-Modified: Fri, 15 Oct 2012 18:36:05 GMT
Etag: "21641b39f4fdcle86dc67e798a320980"
Date: Fri, 15 Oct 2012 19:00:46 GMT
Server: CASTor Cluster 5.0.0
```

READ response for named object

Example response to a READ request for a named object:

```
HTTP/1.1 200 OK
Castor-System-CID: d36dfca69ba7752f4708b1fa9bf9918b
Castor-System-Cluster: cluster.example.com
Castor-System-Created: Fri, 15 Oct 2012 18:37:08 GMT
Castor-System-Name: file.txt
Castor-System-Version: 1287167828.514
Content-Length: 11
Content-Type: application/x-www-form-urlencoded
Last-Modified: Fri, 15 Oct 2012 18:37:08 GMT
Etag: "a896b8e88fe7fc15c9b8f9b2d19e311d"
Date: Fri, 15 Oct 2012 18:45:12 GMT
Server: CASTor Cluster/5.0.0
```

READ response for unnamed objects

Example response to a READ request for an unnamed object.

```
HTTP/1.1 200 OK
Castor-System-Cluster: cluster.example.com
Castor-System-Created: Fri, 15 Oct 2012 18:16:54 GMT
Content-Type: text/html
Content-Length: 645
Cache-Control: no-cache
Expires: Tue, 05 Oct 2012 19:40:23 GMT
Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE
[ application meta-information ]
Date: Wed, 06 Oct 2012 23:27:03 GMT
Server: CASTor Cluster/5.0.0
[ content ]
```

This response means the node receiving the request is returning the requested content in the message body of the response.

READ response for range headers

When a READ request includes one or more [range headers](#), a successful Swarm response includes only the requested bytes range(s). In this case, instead of returning an HTTP 200 OK response, Swarm returns an HTTP 206 Partial Content response, indicating that only part of the content is being returned. The application-meta-information and content-stream are the same.

```
HTTP/1.1 206 Partial Content
Date: Wed, 1 Sept 2012 15:59:02 GMT
Server: CASTor Cluster 5.0.0
Content-Length: 500
[ application-meta-information ]
CRLF
[ content ]
```

READ response for moved permanently

The following response shows that the requested object was located, but another node in the cluster supplies the content. Additionally, all future requests of this storage cluster should be made through the new access node until another HTTP 301 response is received.

There is no message-body, so the content length is always 0. The value of the **Location** header indicates which node in the cluster receives the redirect.

The client is expected to send another READ request using the exact URI in the Location header.

```
HTTP/1.1 301 Moved Permanently
Date: Wed, 1 Sept 2012 15:59:02 GMT
Server: CASTor Cluster/5.0.0
Location: http://cluster-ip/name-or-uid?auth=2096EFA659295BBB819D1FECCE77D2EF
Content-Length: 0
```

READ response for moved temporarily

The following response is similar to the HTTP 301 response, except the client should continue to use the current node (the one generating this response) for future requests until further notice.

```
HTTP/1.1 307 Temporary Redirect
Date: Wed, 1 Sept 2012 15:59:02 GMT
Server: CASTor Cluster/5.0.0
Connection: close
Location: http://cluster-ip/name-or-uid?auth=2096EFA659295BBB819D1FECCE77D2EF
Content-Length: 0
```

SCSP INFO

This section provides general information about SCSP INFO that applies to both named and unnamed objects.

- [INFO for named objects](#)
- [INFO for unnamed objects](#)
- [INFO for alias objects](#)
- [Normal responses to INFO](#)
- [Error responses to INFO](#)

INFO is a request to the storage cluster to provide information about a specific object. The INFO message is identical in semantics to the READ request, except that the object (if found) is not returned in the response. If the referenced content is found in the cluster, only the meta-information about that object is returned in the form of response headers. The INFO request is formatted as an HTTP request using the HEAD method.

SCSP Method	HTTP Method	RFC 7231 Section
INFO	HEAD	4.3.2



Note

SCSP allows HEAD requests with mismatched `Accept-Encoding` headers to receive responses as if the encoding matched that of the object. Swarm relaxed this RFC restriction because a HEAD request returns no body contents, so there is nothing to encode.

INFO for named objects

```
HEAD /mybucket/samplefile.txt HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
```

INFO for unnamed objects

```
HEAD /06eec5e2c3f1aadcb41ef7fd52adc049 HTTP/1.1
User-Agent: Swarm Client/0.1
```

INFO for alias objects

To INFO an alias object, optionally add the `alias=yes` query argument to the URI portion of the HTTP request line, as shown below.

```
HEAD /41A140B5271DC8D22FF8D027176A0821?alias=yes HTTP/1.1
User-Agent: Swarm Client/0.1
```

Normal responses to INFO

See [SCSP Headers](#) for a list of response headers.

The responses described in this section may be returned by the storage cluster in the case where the requested content was found. The content's meta-information can be returned directly by the node that received the request or the request may be redirected to another node in the cluster.

Note

Different Content-Length values in the responses display if `?checkIntegrity` is added to HEAD and GET requests for the same object. This occurs because the HEAD response returns the Content-Length of the manifest rather than the object.

The following response indicates the requested object was located, but another node in the cluster services the request for meta-information. Additionally, all future requests of this storage cluster should be made through the new access node until another 301 response is received. There is no message-body, so the content length is always 0. The value of the Location header indicates which node in the cluster should receive the redirect.

301 Moved Permanently

The client is expected to send another INFO request using the exact URI contained in the Location header.

```
HTTP/1.1 301 Moved Permanently
Date: Wed, 1 Sept 2010 15:59:02 GMT
Server: CASTor Cluster/v8b2
Connection: close
Location: http://node-ip/name-or-uuid?auth=2096EFA659295BBB819D1FECCE77D2EF
Content-Length: 0
```

307 Temporary Redirect

The following response (307) is similar to the 301 response, except the client should continue to use the current node (the one generating this response) for future requests until further notice.

```
HTTP/1.1 307 Temporary Redirect
Date: Wed, 1 Sept 2010 15:59:02 GMT
Server: CASTor Cluster/5.0.0
Location: http://node-ip/name-or-uuid?auth=2096EFA659295BBB819D1FECCE77D2EF
Content-Length: 0
```

Normal responses for named objects

INFO response for a named object in a domain:

```
HTTP/1.1 401 Unauthorized
  WWW-Authenticate: Digest realm="cluster.example.com/_administrators",
  nonce="05d0a60eelf44361f449496505e05116", opaque="fd9c8e14e20fb7c13408c049b7d222af",
  stale=false,
  qop="auth",
  algorithm=MD5
  WWW-Authenticate: Basic realm="cluster.example.com/_administrators"
  Content-Length: 51
  Content-Type: text/html
  Date: Sat, 16 Oct 2010 00:23:24 GMT
  Server: CASTor Cluster 5.0.0
  Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE
HTTP/1.1 200 OK
  Castor-System-Alias: ec87e3c7c410cc04fc4c838061898d9c
  Castor-System-CID: ffffffffffffffffffffffffffffffffff
  Castor-System-Cluster: cluster.example.com
  Castor-System-Created: Fri, 15 Oct 2010 23:59:40 GMT
  Castor-System-Name: cluster.example.com
  Castor-System-Owner: admin@CASTor administrator
  Castor-System-Version: 1287187180.959
  Content-Length: 0
  Last-Modified: Fri, 15 Oct 2010 18:35:56
  GMT lifepoint: [] reps=16
  Etag: "da8bfbb04d089b9c22ae77747f327233"
  Date: Sat, 16 Oct 2010 00:23:24 GMT
  Server: CASTor Cluster/5.0.0
```

The initial 401 Unauthorized response is a normal initial response to HTTP authentication. An initial 401 on an INFO on a domain is returned because access to a domain always requires administrator credentials.

```
HTTP/1.1 200 OK
  Castor-System-Alias: d36dfca69ba7752f4708b1fa9bf9918b
  Castor-System-CID: ec87e3c7c410cc04fc4c838061898d9c
  Castor-System-Cluster: cluster.example.com
  Castor-System-Created: Fri, 15 Oct 2010 18:36:05 GMT
  Castor-System-Name: bucket
  Castor-System-Version: 1287167765.255
  Content-Length: 0
  Content-Type: application/x-www-form-urlencoded
  Last-Modified: Fri, 15 Oct 2010 18:36:05 GMT
  Etag: "21641b39f4fdcle86dc67e798a320980"
  Date: Fri, 15 Oct 2010 23:54:44 GMT
  Server: CASTor Cluster/5.0.0
HTTP/1.1 200 OK
  Castor-System-CID: d36dfca69ba7752f4708b1fa9bf9918b
  Castor-System-Cluster: cluster.example.com
  Castor-System-Created: Fri, 15 Oct 2010 22:09:19 GMT
  Castor-System-Name: file.txt
  Castor-System-Version: 1287180559.436
  Content-Length: 26
  Content-Type: application/x-www-form-urlencoded
  Last-Modified: Fri, 15 Oct 2010 22:09:19 GMT
  Etag: "c744aa90d375aa3e1f228f74b7960e54"
  Date: Fri, 15 Oct 2010 23:51:44 GMT
  Server: CASTor Cluster/5.0.0
```

The response for a named object is very similar to the response for a bucket except that Castor-System-CID is the identifier of the named object's parent (the bucket).

Normal responses for unnamed objects

The following response indicates that the node that received the request found the requested content and is returning meta-information about the object in the headers of this response.

```
HTTP/1.1 200 OK
Date: Wed, 1 Sept 2010 15:59:02 GMT
Server: CASTor Cluster/5.0.0
Content-Type: image/jpeg
Content-Length: (length of the content of the Swarm object)
Replica-Count: (number of replicas in cluster)
[ application-meta-information ]
```

Error responses to INFO

The responses in this section may be returned by the storage cluster when the specified content cannot be found or there is a problem with the INFO request.

The following response indicates a problem with the INFO request, such as missing mandatory headers, invalid message body, or any other violation of HTTP/1.1 by the HEAD request. The reason for the error is included in the status line.

```
HTTP/1.1 400 Bad Request
Date: Wed, 1 Sept 2010 15:59:02 GMT
Server: CASTor Cluster 5.0.0
Content-Length: 24
Content-Type: text/html
```

Indicates the requested object cannot be located in this cluster:

```
HTTP/1.1 404 Not Found
Date: Wed, 1 Sept 2010 15:59:02 GMT
Server: CASTor Cluster 5.0.0
Content-Length: 56
Content-Type: text/html
```

- [Getting Node Status and Cluster Capacity](#)
- [Getting Replica Counts and Location](#)

Getting Node Status and Cluster Capacity

An INFO request submitted without a domain, a name, or a UUID returns basic status information:

- Counts for high-level node methods (such as READ, WRITE, DELETE)
- Cluster-wide values for **Space Available** and **Total Capacity**.

```
HEAD / HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
```

 **Tip**
 This information is also available in the [Swarm Storage UI](#)

Getting Replica Counts and Location

Use the query argument **countreps=yes** to request INFO to return the number and location of object replicas that are online in the cluster. This was previously the default behavior but was changed as of 4.0 replicas are not counted unless requested.

Include the **countreps=yes** query argument to enable Swarm to return the replica count and location of an object:

```
HEAD /mybucket/samplefile.txt?countreps=yes HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
```

Swarm returns the replica count and location when completed.

```
HEAD / HTTP/1.1 HTTP/1.1 200 OK
Castor-System-Cluster: cluster.example.com
Castor-System-Created: Sun, 07 Jul 2013 17:10:06 GMT
Content-Length: 4
Content-Type: application/x-www-form-urlencoded
Last-Modified: Sun, 07 Jul 2013 17:10:06 GMT
Location: http://192.168.1.6:80/70ef3152c831c2c80bbce6505dfb7d0a
Volume: 992c9259b37637927cec444bf9865b8c
Location: http://192.168.1.52:80/70ef3152c831c2c80bbce6505dfb7d0a
Volume: 4d2ffe4f5b8403af3bb9b5408c1babf7
Replica-Count: 2
Etag: "70ef3152c831c2c80bbce6505dfb7d0a"
Volume: 992c9259b37637927cec444bf9865b8c
Volume-Hint: 4d2ffe4f5b8403af3bb9b5408c1babf7
Entity-MD5: 5r0hE+hjVdcj6owxoDRhaw==
Stored-Digest: e6bd2113e86355d723ea8c31a034616b
MinReps: 2
Date: Sun, 07 Jul 2013 17:10:33 GMT
Server: CASTor Cluster/6.1.0
```

SCSP WRITE

- [Write for contexts](#)
- [S3 compatibility](#)
- [WRITE for named objects](#)
 - [WRITE that overwrites object](#)
 - [WRITE that prevents overwriting](#)
 - [Using PUT Create for named objects](#)
 - [Exception](#)
 - [Preventing overwriting: If-None-Match](#)
 - [Note](#)
 - [Note](#)
- [WRITE for unnamed objects](#)
 - [WRITE unnamed to host domain](#)
- [WRITE for alias objects](#)
 - [WRITE for alias object](#)
- [WRITE for erasure-coded objects](#)
 - [Note](#)
- [WRITE for large files \(Expect: 100-continue\)](#)
 - [Error if Expect header is missing](#)

This section provides general information about SCSP WRITE that applies to both named and unnamed objects.

WRITE is a request to the storage cluster to create a new object. The WRITE request is formatted as an HTTP request using the POST method.

SCSP Method	HTTP Method	RFC 7231 Section
SCSP WRITE	POST	4.3.3

Write for contexts

The Swarm setting [scsp.requireExplicitContextCreate](#) protects content-bearing objects from being created erroneously as contexts (buckets or domains). With this setting enabled, Swarm does not create a context object unless it includes the required header: Content-type: application/castorcontext. (v9.1)

S3 compatibility

The Swarm setting [scsp.autoContentMD5Computation](#) improves S3 compatibility by automating Content-MD5 hashing. the gencontentmd5 query argument or the deprecated Expect: Content-MD5 header does not need to be included on writes (although a separate Content-MD5 header may want to be supplied for content integrity checking). This setting is ignored wherever it is invalid, such as on a multipart initiate/complete or an EC APPEND. (v9.1)

WRITE for named objects

The existing object is overwritten with a new version if performing a WRITE of a named object that already exists.

WRITE that overwrites object

```
POST /bucket/photo.jpg HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
Content-Length: 43402
Expect: 100-continue
Content-Type: image/jpeg
Content-Language: en/us, x-pig-latin
Content-Version: 42
CRLF
[ content ]
```

Include the **If-None-Match: *** request header to prevent overwriting an existing object.

- Swarm WRITES the named object if the named object does not exist.
- Swarm responds with an HTTP 412 Precondition Fail error if the named object exists.

WRITE that prevents overwriting

```
POST /bucket/photo.jpg HTTP/1.1
If-None-Match: *
Host: cluster.example.com
User-Agent: Swarm Client/0.1
Content-Length: 43402
Expect: 100-continue
Content-Type: image/jpeg
Content-Language: en/us, x-pig-latin
Content-Version: 42
CRLF
[ content ]
```

Using PUT Create for named objects

Add the **scsp.allowPutCreate=True** to the [configuration parameters](#) to configure Swarm to allow using the HTTP PUT operation to create new named objects. The [putcreate query argument](#) can also enable it.

Exception

Although domains and buckets are named, Swarm processes all PUT requests on these objects as updates, regardless of the setting.

- Swarm fails the request with a 400 Bad Request error if the [putcreate=yes](#) query argument is used on a domain or bucket.
- Swarm silently ignores it and processes the request as an ordinary PUT if the [scsp.allowPutCreate](#) parameter is enabled.

Preventing overwriting: If-None-Match

In contrast to an unnamed object, the existing object is overwritten with a new version if performing a WRITE of a named object that already exists. Include the **If-None-Match: *** request header to prevent overwriting an existing object.

- Swarm WRITES the named object if the named object does not exist.
- Swarm responds with an HTTP 412 Precondition Fail error if the named object exists.



Note

Swarm returns an HTTP 412 error on SCSP WRITE of named objects if the domain or bucket does not exist or cannot be loaded.



Note

If-None-Match:* can erroneously report that an object exists during the time window after it is flagged for deletion by policy but before it is removed from disk. This window is determined by the HP cycle time.

WRITE for unnamed objects

Swarm makes no assumptions about **User-Agent** (except it is an HTTP/1.1 client). The **Host** header must conform to the requirements of [Section 14.23](#) of the HTTP/1.1 spec.

WRITE unnamed to host domain

```
POST / HTTP/1.1
Host: cluster.example.com
User-Agent:Swarm Client/0.1
Content-Length: 43402
Expect: 100-continue
Content-Type: image/jpeg
Content-Language: en/us, x-pig-latin
Content-Version: 42
CRLF
[ content ]
```

A new object is created and a new UUID is returned if performing a WRITE of an unnamed object. A new object is created and a new alias UUID is returned if performing a WRITE of an alias object.

See [WRITE for Unnamed Objects](#).

WRITE for alias objects

Add **alias=yes** to create **alias** objects:

WRITE for alias object

```
POST /?alias HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
```

WRITE for erasure-coded objects

A new object written to the storage cluster is erasure-coded if it meets the [EC criteria](#) language.

See [Working with Large Objects](#).



Note

Swarm returns a **HTTP 503 Service Unavailable** error if performing a WRITE of an object of more than 4 TB in size.

WRITE for large files (Expect: 100-continue)

The [Expect: 100-continue header](#) tells the server the client waits after sending the header lines and before sending the content in the message body. The Swarm server can respond with a redirect or an error response.

- The server returns an **HTTP 100 Continue** response, telling the client to transmit the entity body if the server is ready to store the contents. The client needs to wait for a 100 Continue response from the server before proceeding to send the data.
- The server sends an **HTTP 413 Request entity too large** error response and closes the connection if the server is not ready.

Swarm allows the client to omit the Expect: 100-continue header, sending all content at once. The server reads and discards all data if it must respond with a redirect or error. Swarm logs a warning for WRITE messages that include more than 65536 bytes:

Error if Expect header is missing

Please use `Expect: 100-continue` for large amounts of data.

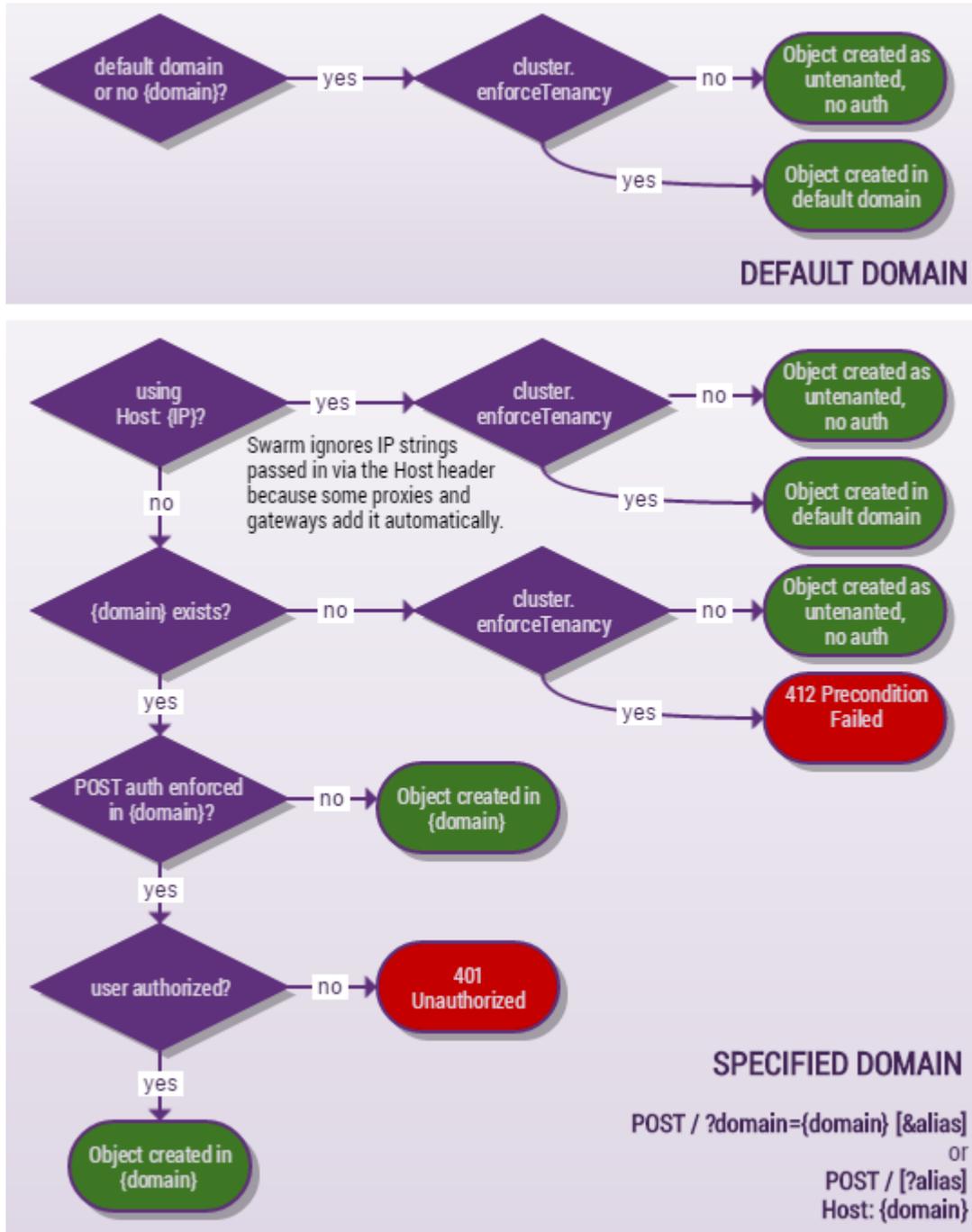
The cluster returns an **HTTP 507 Insufficient Storage** error if any node in the cluster does not have enough space to write the object.

- [How enforceTenancy Works](#)
- [Error Responses to WRITE](#)
- [Normal Responses to WRITE](#)
- [WRITE with Replicate ROW](#)
- [WRITE for Unnamed Objects](#)

How enforceTenancy Works

Before Swarm creates an unnamed object in a domain, it checks the value of the `cluster.enforceTenancy` configuration setting and then performs a specific set of procedures, depending on whether the cluster administrator enabled `cluster.enforceTenancy`, which is disabled by default.

The following figure summarizes how the `cluster.enforceTenancy` setting affects the writing of unnamed objects:



i Note

Regardless of the value of the [enforceTenancy setting](#), no domain specification is needed or recognized for GET, HEAD, or DELETE requests on unnamed objects, whether alias or immutable.

Error Responses to WRITE

Errors are accompanied by three error headers (v9.1):

- `CastorSystemErrorCode` - The request error code (if applicable). This code is usually a 4xx or 5xx HTTP response code, but it may not match the response code on the request.
- `CastorSystemErrorText` - The request error description (if applicable). Provides a human-readable description of the error.
- `CastorSystemErrorToken` - A unique error token for the specific error path (if applicable). Provides a parsed token that uniquely identifies the error.

The storage cluster can return the following responses when the content length header does not match the actual content length, the specified content cannot be written to the cluster, or if there is a problem with the WRITE request itself.

<p>400 Bad Request</p>	<p>The response below indicates a problem with the WRITE request, such as missing mandatory headers, invalid message body, or any other violation of HTTP/1.1 by the POST request. The reason for the error is included in the message body of the response.</p> <pre>HTTP/1.1 400 Bad Request Date: Wed, 1 Sept 2012 15:59:02 GMT Server: CASTor Cluster/5.0.0 Content-Type: text/html Content-Length: 24 CRLF Host header is required.</pre>
<p>411 Length Required</p>	<p>The response below indicates the WRITE request did not supply the actual content-length, which is required.</p> <pre>HTTP/1.1 411 Length Required Date: Wed, 1 Sept 2012 15:59:02 GMT Server: CASTor Cluster/5.0.0 Content-Type: text/html Content-Length: 93 CRLF WRITE requests must include a Content-Length header specifying the exact byte-length of the content to be stored.</pre>
<p>412 Precondition Failed</p>	<p>The response indicates the named object already exists or the bucket or domain cannot be found.</p>
<p>507 Insufficient Storage</p>	<p>The response below indicates the WRITE request did not succeed because the storage cluster did not have sufficient resources to fulfill it. The specific resource constraint is described in the entity (message-body) of the response.</p> <pre>HTTP/1.1 507 Insufficient Storage Date: Wed, 1 Sept 2012 15:59:02 GMT Server: CASTor Cluster/5.0.0 Content-Type: text/html Content-Length: 38 CRLF Not enough disk space to store requested content.</pre>

Normal Responses to WRITE

- [Note](#)

See [SCSP Headers](#) for a list of response headers.

The responses listed here can be returned by the storage cluster when the new object can be created as requested. The content can be created and written to the cluster by the node that receives the request or redirected to another node in the cluster.

Note

The `Replica-Count` header indicates the number of synchronous replicas created, if the number is greater than 1.

<p>201 Created</p>	<p>The response below indicates the storage cluster has stored at least one copy of the supplied object. If required, other nodes can store additional replicas at a later time. The value of the <code>Location</code> header provides the URI where the newly-created object can be accessed in the cluster. The last portion of the URI is the symbolic name or UUID of the object.</p> <p>For unnamed objects, this value is repeated as the value of the <code>Content-UUID</code> header field. Note: the <code>Content-Type</code> and <code>Content-Length</code> headers refer to the message payload of the response (if any).</p> <pre>HTTP/1.1 201 Created Date: Wed, 1 Sept 2015 15:59:02 GMT Server: CASTor Cluster/5.0.0 Location: http://node-ip/<bucket/name uuid> Content-UUID: 41a140b5271dc8d22ff8d027176a0821 Content-Type: text/html Content-Length: 68 CRLF A new object has been created as requested. Its URL is http://node-ip/<bucket/name uuid></pre>
<p>202 Accepted</p>	<p>The response occurs for certain types of writes, such as a multi-part completion. It means that the request was accepted, and it is pending. The final status is returned both in the body, and as a trailing header named <code>castor-system-result</code>.</p> <p>After a 202, the request may still fail.</p>
<p>301 Moved Permanently</p>	<p>The response indicates the request has been redirected to another node, and should be retried there. All future requests of this storage cluster should be made through the new access node until another 301 response is received. There is no message-body, so the content length is always 0. The value of the <code>Location</code> header indicates which node in the cluster receives the redirect.</p> <p>The client is expected to send another POST request using the exact URI contained in the <code>Location</code> header, including the <code>auth=</code> query argument.</p> <pre>HTTP/1.1 301 Moved Permanently Date: Wed, 1 Sept 2015 15:59:02 GMT Server: CASTor Cluster/5.0.0 Location: http://node-ip/<bucket/name uuid>?auth=value Content-Length: 0</pre>

307 Temporary Redirect

The response below is similar to the 301 response, except that the client should continue to use the current node (the one generating this response) for future requests until further notice.

```
HTTP/1.1 307 Temporary Redirect
Date: Wed, 1 Sept 2015 15:59:02 GMT
Server: CAStor Cluster/5.0.0
Location: http://node-ip/<bucket/name | uuid>/?auth=B1E1509329C7A5DD90DCF6642DFE
Content-Length: 0
```

WRITE with Replicate ROW

- [Success conditions for ROW](#)
 - [Note](#)
- [Implementing ROW](#)
- [replicate argument](#)
- [Replica-Count header](#)
- [Responses for replicating objects](#)

The Replicate on Write (ROW) option forces Swarm to write a new object to one or more additional nodes before returning a success status. Using this content protection option, verify two or more object replicas (instances) exist in the cluster before the client write request is completed.

Success conditions for ROW

These are the success conditions for a ROW request:

- A POST on an immutable object creates at least two replicas.
- Any write operation for an alias object, or a POST for a named object.



Note

The reason for treating named objects like existing alias objects is they may already exist. Allowing these writes to succeed with one replica verifies no old versions can be inadvertently deleted by the HP if the request fails with one replica.

Implementing ROW

Implement ROW in these ways:

- **Globally**, set the configuration file to enabling (*recommended*) or disabling cluster-wide ROW.
 - See [Configuring Replicate On Write](#).
- **Programmatically**, use a replicate query argument when needing to override the cluster-wide ROW configuration.
- **Creating or updating a bucket**. The replicate=immediate option quickly invalidates cached bucket versions in the cluster so the latest version is implemented in the cluster. It also prevents subsequent permission errors because out-of-date permissions are used from the prior version.

```
curl -i
  --post301
  --data-binary ''
  --location-trusted 'http://172.16.0.35/bucket?domain=test.example.com&replicate=immediate'
  -D create-bucket.log
```

replicate argument

Add the replicate query argument to control how Swarm implements ROW on a given request.

Use this argument to limit or disable ROW for the request if cluster-wide ROW is **enabled** (*recommended*):

- **replicate=x** (where *x* is an integer) creates *x* replicas on write. For example, replicate=1 allows the write to succeed with one instance of the object.

Use these arguments to enable ROW for the request if cluster-wide ROW is **disabled**:

- **replicate=immediate** is replicate=2, which verifies two replicas are written.
- **replicate=full** is replicate={# of reps specified by lifepoint, or else `policy.replicas default`}

The number of replicas Swarm makes synchronously on the request cannot exceed the number of replicas specified in the lifepoint (or, if none, `policy.replicas default`) in every case. A request with replicate=3 causes 2 replicas to be synchronously created on the request for an object with no lifepoint specified and a cluster with default=2.

Replica-Count header

Swarm indicates the number of replica created with the request in the **Replica-Count** header. Check the header value in the response to verify the correct number of replicas is received.

Swarm returns an **HTTP 412 Preconditioned Failed** response if Swarm cannot locate at least two nodes in the cluster that can replicate the object. Swarm proceeds with the request if Swarm can locate a PAN and one ROW peer node.

A ROW request can return successfully with one replica created, it does not attempt to perform the operation if it cannot find at least two nodes up front.

Responses for replicating objects

Swarm returns an **HTTP 412 Preconditioned Failed** response if Swarm is replicating an object and the cluster cannot locate at least two nodes to store the replicas initially.

Swarm returns an **HTTP 201 Created** response if Swarm locates one node to store the replica. Applications that need to verify the requested number of created replicas should check the **Replica-Count** header value to verify how many replicas are created in the cluster.

Repeat the request if the requested number of replicas does not match the **Replica-Count** header value. The Health Processor creates the additional replicas at a later time if this condition is not met.

Swarm locates two peer nodes—including the SAN—to perform the write to POST any unnamed object. An immutable POST is considered a success if at least two replicas complete successfully when two nodes are found and the writes are initiated. The write fails and Swarm returns an **HTTP 412 Preconditioned Failed** response if Swarm cannot locate two peer nodes. All other writes are considered a success if at least one replica completes successfully.

WRITE for Unnamed Objects

SCSP methods can be created and run on unnamed objects in any domain. Housing (*tenanting*) unnamed objects in domains supports metered environments that need to allocate storage to users based on the domain. The domain is not used to later locate the unnamed object in the cluster.

A cluster administrator must create the domain and enabled the cluster configuration setting [cluster.enforceTenancy](#) to be able to create an unnamed object in a domain. See [How enforceTenancy Works](#).

An unnamed object is written to a specific domain by including the domain in a query argument or in the HOST header:

Which domain?	Alias Object	Immutable Object	
Unspecified (default domain)	POST /?alias	POST /	<p>Every unnamed object that has no domain explicitly defined belongs to the default cluster domain.</p> <p>Verify the cluster administrator has set up a default domain, which is a domain name that exactly matches the name of the cluster.</p>
By query argument	POST /?domain=domain-name&alias	POST /?domain=domain-name	
By host name	POST /?alias Host: domain-name	POST / Host: domain-name	<div style="border: 1px solid #ccc; padding: 10px;"> <p>Performance Warning</p> <p>Swarm performs several attempts to look up the invalid domain before timing out on every request if the application passes an invalid HOST header: a domain that does not exist in or matches the cluster name.</p> </div>

SCSP DELETE

- [DELETE for named objects](#)
 - [Guidelines for DELETE](#)
 - [DELETE for domains and buckets](#)
 - [Note](#)
 - [Reusing bucket names](#)
 - [Best practice](#)
- [DELETE for unnamed objects](#)
 - [Note](#)
 - [Note](#)
- [DELETE for alias objects](#)

This section provides general information about SCSP DELETE that applies to both named and unnamed objects.

DELETE is a request to the storage cluster to remove a specific object. The DELETE request is formatted as an HTTP request using the DELETE method.

SCSP Method	HTTP Method	RFC 7231 Section
SCSP DELETE	DELETE	4.3.5

DELETE for named objects

```
DELETE /bucket/photo.jpg HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
```

Guidelines for DELETE

- **For bucket requests, use a separate initialization or setup routine that runs less frequently.** Swarm is optimized for calls on individual objects, not domains or buckets (which are centralized resources), so do not make bucket calls on the high-availability code path of a client application.
- **Reuse object names.** After a named object is deleted, another object with the same name can be created in the same bucket. Unlike unnamed objects, whose UUIDs are not reused, names *can* be reused.
- **Pause before recreating.** Deleting a named object involves an underlying update, for Swarm to write a special marker value to the name. When recreating a named object after deleting it, wait at least one second.

DELETE for domains and buckets

Delete any objects contained within a domain or bucket when deleting them or else these objects are orphaned, lost, and consume disk space unnecessarily. These deletes are recursive, iterating until every object contained in the domain or bucket is dealt with.

 **Note**

Swarm generates an SCSP error if attempting to delete a domain or bucket without having a recursive argument or parameter in force.

The [recursive query argument](#) must be included to delete a domain or bucket:

- **recursive | recursive=true|yes**
 - Grants a 1-week grace period (default) during which the domain or bucket can be restored before the health processor begins reclaiming the space.
 - Edit the [health.recursiveDeleteDelay parameter](#) to change the length of the grace period
- **recursive=now**
 - Grants no grace period. The health processor begins reclaiming the space immediately.

Avoid changing the **recursive** argument if existing integrations do not use the argument. Add a global [configuration parameter](#): **scsp.autoRecursiveDelete=True**.

An erroneously deleted domain or bucket can be restored without data loss if it is within the grace period. See [Restoring Domains and Buckets](#).

Reusing bucket names

A bucket can be deleted and another bucket with the same name can be recreated:

- The new bucket is a different bucket that happens to have the same name.
- All objects in a bucket are inaccessible after deleting it, even if another bucket with the same name is subsequently created.

Best practice

Wait at least *twice* the value of **cache.realmStaleTimeout** before attempting to recreate a bucket with the same name as a bucket deleted: the default is 600 seconds (10 minutes), so wait 20 minutes, then create the new bucket. This waiting period applies to reusing names of *buckets*: deleting a named object and recreating an object with that name requires a 1-second pause.

DELETE for unnamed objects

```
DELETE /7A25E6067904EAC8002498CF1AE33023 HTTP/1.1
User-Agent: Swarm Client/0.1
```

The content associated with the name or UUID supplied in the request is no longer available if the method succeeds. This does not imply all copies of the content are erased. The cluster now responds to any READ request for that UUID with an HTTP 404 Not Found error.

Note

Swarm deletes both the manifests *and* the segments of an erasure-coded object. Erasure coding allows storage of larger objects with a smaller footprint. See [Working with Large Objects](#).

An object can be deleted by an application or lifecycle as follows:

- **Application deleting an object** - All online replicas in a single cluster are removed immediately after a delete method is executed on an object. (An *online replica* is one that resides on a cluster node online at the time the delete is issued.) The cluster recalls the name or UUID has been deleted for 14 days, in the event that one or more nodes holding replicas of the deleted object are off line at the time the delete was issued.

- Policy deleting an object** - An object can have a *storage policy* defined by the application and stored along with it. Part of the storage policy may be an expiration period, beyond which the object is to be removed. In the case of a policy-defined deletion, all replicas, wherever they are stored, are deleted at approximately the same time and become unavailable at most one second after the expiration date and time.


Note

The UUID of a deleted object is not reused, even if the object is mutable.

DELETE for alias objects

To delete **alias** objects, add a query argument **alias**:

```
DELETE /7A25E6067904EAC8002498CF1AE33023?alias HTTP/1.1
User-Agent: Swarm Client/0.1
```

Normal Responses to DELETE

- [DELETE response](#)
- [DELETE response for moved permanently](#)
- [DELETE response for moved temporarily](#)

See [SCSP Headers](#) for a list of response headers.

DELETE response

The following response indicates that the content was found and is being deleted:

```
HTTP/1.1 200 OK
Castor-System-Cluster: cluster.example.com
Castor-System-Created: Fri, 15 Oct 2010 18:16:54 GMT
Content-Type: application/x-www-form-urlencoded
Last-Modified: Fri, 15 Oct 2010 18:16:54 GMT
Etag: "06eec5e2c3flaadcb41ef7fd52adc049"
Content-Length: 0
Date: Fri, 15 Oct 2010 21:43:51 GMT
Server: CASTor Cluster/5.0.0
```

DELETE response for moved permanently

The following response indicates the content can be deleted as requested, but another node in the cluster completes the Delete. Additionally, all future requests of this storage cluster should be made through the new access node until another 301 response is received. The value of the Location header indicates which node in the cluster should receive the redirect. The client is expected to send another DELETE request using the exact URI contained in the Location header.

```
HTTP/1.1 301 Moved Permanently
Date: Wed, 1 Sept 2010 15:59:02 GMT
Server: CASTor Cluster/5.0.0
Location: http://node-ip/name-or-uuid?auth=2096EFA659295BBB819D1FECCE77D2EF
Content-Length: 0
```

DELETE response for moved temporarily

The following response indicates the content can be deleted as requested, but another node in the cluster completes the Delete. All future requests of this storage cluster should be made through the new access node until another 301 response is received. The value of the Location header indicates which node in the cluster is assigned the redirect. The client is expected to send another DELETE request using the exact URI contained in the Location header.

```
HTTP/1.1 307 Temporary Redirect
Date: Wed, 1 Sept 2010 15:59:02 GMT
Server: CASTor Cluster 5.0.0
Location: http://node-ip/name-or-uuid?auth=2096EFA659295BBB819D1FECCE77D2EF
Content-Length: 0
```

This response is similar to the 301 response, except the client should continue to use the current node (the one generating this response) for future requests until further notice.

Error Responses to DELETE

The storage cluster can return the following responses when the specified content cannot be deleted from the cluster or there is a problem with the DELETE request.

<p>400 Bad Request</p>	<p>The following response indicates a problem with the DELETE request, such as missing mandatory headers, invalid message body, or any other violation of HTTP/1.1 by the DELETE request. The reason for the error is included in the message body of the response.</p> <pre>HTTP/1.1 400 Bad Request Date: Wed, 1 Sept 2010 15:59:02 GMT Server: CASstor Cluster/5.0.0 Connection: close Content-Type: text/html Content-Length: 24 CRLF Host header is required.</pre>
<p>404 Not Found</p>	<p>The following response indicates the requested object cannot be located in this cluster.</p> <pre>HTTP/1.1 404 Not Found Date: Wed, 1 Sept 2010 15:59:02 GMT Server: CASstor Cluster/5.0.0 Content-Length: 56 Content-Type: text/html CRLF The requested resource is not available in this cluster.</pre>
<p>403 Forbidden</p>	<p>The following response indicates the requested object cannot be deleted because its current state forbids it. An object's lifecycle may include periods of time when users are not allowed to delete it from the cluster.</p> <pre>HTTP/1.1 403 Forbidden Date: Wed, 1 Sept 2010 15:59:02 GMT Server: CASstor Cluster/5.0.0 Content-Length: 56 Content-Type: text/html CRLF The requested resource cannot be deleted at this time.</pre>

SCSP UPDATE

- [Special Query Arguments](#)
- [UPDATE for named objects](#)
- [UPDATE for unnamed objects](#)
 - [Note](#)
- [UPDATE for alias objects](#)
- [Normal Responses to UPDATE](#)
- [Error Responses to UPDATE](#)
 - [Rapid updates](#)

This section provides general information about SCSP UPDATE applying to both named and unnamed objects.

The UPDATE request is formatted as an HTTP request using the PUT method.

SCSP Method	HTTP Method	RFC 7231 Section
SCSP UPDATE	PUT	4.3.4

Special Query Arguments

replicate	Protects rapid updates	Important: Objects can be updated at a maximum frequency of <i>once per second</i> . Updating more frequently can cause unpredictable results with the stored object version and can trigger an HTTP 409 (Conflict) error. Include the <code>replicate=immediate</code> query argument to verify more than one node returns the latest version in a subsequent read if an application updates objects faster than once per second.
newname	Renames object	Use the newname query argument to rename a named object within the same bucket, which provides a new name with the update request (PUT, COPY, APPEND). Requests for the original name return an HTTP 404 Not Found and the prior search metadata is removed after an object is renamed. Note: the <code>newname</code> argument also allows renaming domains and buckets.
preserve	Updates custom headers	PUT only saves new headers, but the <code>preserve</code> argument allows keeping the existing headers as well as save any new ones. (v9.5)

UPDATE for named objects

UPDATE is a request to the storage cluster to modify a specific named object or alias object with new content. The UPDATE request is formatted as an HTTP request using the [PUT](#) method:

```
PUT /bucket/file.txt HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
Content-Length: 43402
Expect: 100-continue
Content-Type: image/jpeg
Content-Language: en/us, x-pig-latin
Content-Version: 42
Last-Modified: Wed, 1 Sept 2010 15:59:02 GMT
Created-Date: Wed, 1 Sept 2010 15:59:02 GMT
CRLF
[ content ]
```

UPDATE for unnamed objects

The UPDATE request is formatted as an HTTP request using the PUT method. The normal response to a PUT request, similar to a POST, is an HTTP 201 Created response.

```
PUT /06eec5e2c3flaadcb41ef7fd52adc049 HTTP/1.1
Host: cluster.example.com
User-Agent: Swarm Client/0.1
Content-Length: 43402
Expect: 100-continue
Content-Type: image/jpeg
Content-Language: en/us, x-pig-latin
Content-Version: 42
Last-Modified: Wed, 1 Sept 2010 15:59:02 GMT
Created-Date: Wed, 1 Sept 2010 15:59:02 GMT
CRLF
[ content ]
```

PUT returns an HTTP 404 Not Found error if the object name or UUID you specify in the command does not exist.

Note

The object can be erasure-coded, which has a smaller storage footprint, if a non-erasure-coded object is updated and the update causes the object to meet the criteria described in [Erasure Coding](#).

UPDATE for alias objects

Adding **alias=yes** is optional to update **alias** object because this method applies only to mutable objects.

The content sent in the body of the request is written as a new object if UPDATE succeeds on an alias object. The first line of the HTTP PUT is updated to point to the new object, and the original UUID of the object is returned to the client.

Normal Responses to UPDATE

See [SCSP Headers](#) for a list of response headers.

Error Responses to UPDATE

Swarm responds with an **HTTP 409 Conflict** if the UPDATE method is executed on an object in a domain but the domain does not exist or is not in the content cache on the node that receives the request.

Rapid updates

Rapid updates of an object can trigger an HTTP 409 Conflict error, a "Later version already exists."

Rapid updates or overwrites to an object in a *versioned bucket* can cause temporary listing inconsistency, even when `replication=immediate` is used (default with Gateway). Those unlisted versions are accessible directly by the `versionid`. Using a 1-second delay should avoid this.

SCSP APPEND

This section provides general information about SCSP APPEND that applies to both named and unnamed objects.

- [Special Query Arguments](#)
- [Guidelines for APPEND](#)
- [APPEND for named and alias objects](#)
 - [Example APPEND for named object](#)
 - [APPEND for unnamed objects](#)
- [Normal responses to APPEND](#)
- [Error responses to APPEND](#)

APPEND is a request to the storage cluster to append arbitrary content data onto the end of an existing named object or aliased object while maintaining the previously populated metadata and object name or alias UUID. No whitespace or other characters are inserted by Swarm between the original and appended data. APPEND is not valid for immutable unnamed objects.

- **Replicated** – For replicated objects, the original content data is copied by Swarm from the original object and then the data supplied in the request is appended to it. Data appended to a replicated object can cause the object to become erasure-coded if the additional appended data pushes the object size over the configured [policy.ecMinStreamSize](#) threshold.
- **Erasure-coded** – APPEND requests for previously erasure-coded objects with version 6.0 are optimized to write a new set of segments with the appended data and update the manifest, instead of rewriting the whole object to include the appended data as with replicated objects. The request fails if the EC constraints cannot be met on the APPEND request. For example, if encoding is 5:2 and there are only six nodes, the APPEND request fails.

Special Query Arguments

replicate	Protects rapid updates	Important: Objects can be updated at a maximum frequency of <i>once per second</i> . Updating more frequently can cause unpredictable results with the stored object version. Include the <code>replicate=immediate</code> query argument to guarantee more than one node can return the latest version in a subsequent read if an application updates objects faster than once per second.
newname	Renames object	To rename a named object within the same bucket, use the newname query argument , which provides a new name with the update request (PUT, COPY, APPEND). After renaming an object, requests for the original name return a 404 Not Found and the prior search metadata is removed. (Note the <code>newname</code> argument allows renaming domains and buckets.)
preserve	Updates custom headers	APPEND only saves the existing headers, but the <code>preserve</code> argument updates the existing headers with those included on the request, if any. Cannot be used with <code>replace</code> . (v9.5)
replace	Replaces custom headers	APPEND only saves the existing headers, but the <code>replace</code> argument removes the existing headers and saves the new ones included on the request, if any. Cannot be used with <code>preserve</code> . (v9.5)

Guidelines for APPEND

- **Include header for known or unknown size.** Must include either the Content-length or Transfer-Encoding: chunked header.

- Use the **Content-length** header if the size of the object is known. The Content-length value must specify the correct length of the appended content data. The Content-length header in the object is adjusted to reflect the actual length of the original data plus the appended data.
- Use the **Transfer-Encoding: chunked** header (or the UNDETERMINED_LENGTH parameter if using the SDK) if the size of the object is not known (such as a live video feed). This header tells Swarm the size of the appended data is unknown. Do *not* combine this header with the Content-length header. All other headers stored with the object are copied without change to the newly-updated object. The x-acme-meta-* and lifepoint headers in the preceding examples are ignored.
- **Content-MD5 Headers corrected.** Swarm computes the digest of the content data plus the appended data and compares it with the provided MD5 hash if providing a Content-MD5 header with the APPEND request. This assumes either access to the original data or a running digest is maintained to which appended data is added before each APPEND request. A Content-MD5 header is removed when new data is appended to the object if it was persisted with the original object. Any new, correct Content-MD5 supplied with an append is persisted with the new revision and returned on any subsequent GET or HEAD.
- **Omit Range Headers.** Range headers are incompatible with the APPEND method. Including a Range header with an APPEND request results in a 400 Bad Request error response. Other aspects of the APPEND method, including response codes, are the same as PUT.

APPEND for named and alias objects

The syntax of an APPEND request is similar to a [PUT](#). As with PUT, the object name or UUID returned after a successful APPEND matches the one supplied in the request. APPEND for an alias object is the same as for a named object except the object's UUID is used instead of a name on the first line of the command.

Example APPEND for named object

```
APPEND /mybucket/samplefile.txt HTTP/1.1
Host: cluster.example.com
Content-length: 29
x-acme-meta-color: blue
x-acme-meta-weight: 42
lifepoint: [Sunday, 06-Nov-2010 08:49:37 GMT] reps=3, deletable=no
lifepoint: [] delete
Status: Approved
```

APPEND for unnamed objects

APPEND is not supported for unnamed immutable objects: the UUID *must* be an alias object. The query argument **alias=yes** is optional as of v7.0.

Swarm returns a **404 Not Found** error when the object is not alias (appending to an immutable object or to a UUID that does not exist).

Normal responses to APPEND

The APPEND method returns a response only after all data is copied and the update is complete.

See [SCSP Headers](#) for a list of response headers.

Error responses to APPEND

Swarm responds with **HTTP 409 Conflict** if the APPEND method is executed on an object in a domain but the domain does not exist or is not in the content cache on the node receiving the request.

SCSP COPY

This section provides general information about SCSP COPY applicable to both named and alias objects. Immutable unnamed objects cannot have associated metadata updated.

- [Best practice](#)
- [Headers to preserve](#)
- [COPY for named objects](#)
- [COPY for alias objects](#)
- [Normal responses to COPY](#)
- [Error responses to COPY](#)

The COPY method allows updating the metadata on an object after the initial write. COPY allows changing headers without changing content, such as to update permissions or change a lifepoint. The COPY method does *not* create any new objects: rather, it updates the metadata on an existing object, by copying the content verbatim while replacing the metadata.

Best practice

COPY modifies all headers at once, so calling COPY with a new or modified header value has the effect of dropping all other user-supplied headers originally set on the object.

To use COPY correctly, follow this process:

1. HEAD the original object.
2. Grab all writable persisted headers (see "Headers to preserve", below).
3. COPY the object with those header values, with these changes:
 - a. Add any new headers or updated values.
 - b. Omit any headers to be removed.

The requestor decides exactly which headers are written. The COPY method returns a response after the object update is complete.

i Update frequency

Named and alias objects can be updated at a maximum frequency of *once per second*. Updating more frequently can cause unpredictable results with the stored object version. Include the [replicate query argument](#) to verify more than one node can return the latest version in a subsequent read if an application updates objects faster than once per second.

i Gateway transforms

In the Gateway, domain admins can specify header transformations for POSTs, PUTs, and COPYs. A COPY sent through the Gateway is subject to the transform rules, replacing all headers matching the applicable transform rule header names with values in the rules. Gateway discards any headers in a COPY request matching transform rules and updates those headers with current request values for rule Substitution Variables. COPY replaces date and user variables with the current request values rather than the original values on the object.

Headers to preserve

Following are persisted headers typically preserved when adding metadata to an object:

- Allow
- Cache-Control
- Castor-* (except those with System)
- Content-Base
- Content-Disposition
- Content-Encoding
- Content-Language
- Content-Length
- Content-Location
- Content-MD5
- Content-Type
- Expires
- Lifepoint
- Policy-* (except those with Evaluated[-Constrained])
- X-*Meta[-*]



Tip

Add the **preserve** [query argument](#) to the COPY request to verify any custom metadata existing on the object is carried over to the copy. Include the header name with the new value on the request to overwrite an existing value. (v9.2)

See [SCSP Headers](#) for using these headers with COPY.

COPY for named objects

The syntax of a COPY request is similar to an empty PUT request on an alias object.

```
COPY /some/filename HTTP/1.1
Host: cluster.example.com
Content-Length: 0
x-xml-meta-data-color: blue
x-xml-meta-data-weight: 42
x-xml-meta-data: <size>large</size><color>blue</color><specialorder/>
lifepoint: [Sun, 06 Nov 2010 08:49:37 GMT] reps=3, deletable=no

lifepoint: [] delete
```

Renaming – Use the [newname query argument](#), which provides a new name with the update request (PUT, COPY, APPEND), to rename a named object within the same bucket. Requests for the original name return an HTTP 404 Not Found and the prior search metadata is removed after renaming an object. (Note: the **newname** argument also renaming domains and buckets.)

Behavior of COPY

- **Rewrites EC manifest.** COPY rewrites the object manifest rather than creating a new object with new metadata if an erasure-coded object is modified by COPY. (See [Working with Large Objects.](#))

- **Has no content body.** The Content-Length header is optional. The value must be 0 because there is no content body for a COPY request if this header is included.
- **Replaces metadata headers.** Additional headers (such as the two x-*-meta-* headers in the example and the lifepoint headers), replace all existing metadata in the original object. The one exception is the Content-Length header value continues to provide the original length of the content data.
- **Calculates and compares hashes.**
 - *Non-EC object* - Swarm recomputes the digest of the content data as it copies it and compares it with the provided MD5 hash if the client provides a Content-MD5 header with the COPY request. Similar to a WRITE, if the provided and calculated hashes do not match, the operation fails.
 - *EC object* - The request calculates a new MD5 on non-EC objects. On an EC object, the `gencontentmd5` query argument (or the deprecated `Expect: Content-MD5` header) or `content-md5` header is allowed if the existing object already has a content-md5 stored on it. Any new value must match the existing value.
- **Responses.** Other aspects of the COPY method, including response codes, are the same as PUT. The COPY method returns a response after all data is copied and the object update is complete.

COPY for alias objects

The UUID returned after a successful COPY is identical to the UUID supplied in the request, which is similar to a PUT or APPEND request.

The query argument `alias=true` is an optional acknowledgment the method is executed on an existing alias object. The object specified by the included UUID *must* be an alias object in Swarm.

Failure to perform a COPY on an alias object results in an **HTTP 403 (Forbidden)** response.

Normal responses to COPY

A multipart COPY by part operates like a [multipart write completion](#), sending back an HTTP 202 response code and keep-alive characters to prevent client timeouts. (v9.1.2)

See [SCSP Headers](#) for a list of response headers.

Error responses to COPY

Swarm responds with 409 (Conflict) if executing the COPY method on an object in a domain but the domain does not exist or is not in the content cache on the node on which the request is directed.

SCSP SEND

- [Legacy SEND](#)
- [SEND Requests](#)
 - [Query Arguments for SEND](#)
- [SEND Responses](#)
 - [Chunked encoding](#)
 - [Response Headers](#)
- [Example of SEND](#)

The SCSP SEND method applies to both named and unnamed objects. The SEND request allows explicit transmission of a newly written object from a source cluster to a remote one, such as for keeping two clusters immediately synchronized. The feed SEND method works with any feed type as of Swarm 11.2, so it can force synchronous processing of a specific object on one or more of those feeds.

Best practice – Use this feature with a [replication feed](#), which acts as a catch-up mechanism if the intracluster network is down or the SEND command fails.



Legacy SEND

The [legacy behavior of the SEND](#) method (which was limited to legacy replication feeds) is replaced by the following expanded SEND method. The legacy behavior is preserved for backwards compatibility: invoke it by omitting the required “feedid” or “feedtype” query arguments. (v11.2)

SEND Requests

With a SEND request, provide the path or UUID for the Swarm object to be sent to one or more feeds. Swarm checks the destination cluster to verify whether the object already exists there.

Which node to SEND to – Content Gateway determines the optimal target node for the request if using SEND through Content Gateway (*under development for future release*); select the node if going direct to Storage. All replicas must perform feed processing, but, on a new write, one replica begins replication and S3 backup processing. A SEND request has the best chance of arriving with the replication already in progress, which speeds completion if pointing SEND to the optimal node (which holds this first replica). Determine whether the request is an EC write and whether the request is a multipart completion to find the optimal SEND node. Identify an EC write by the “Manifest: ec” response header.

- Use the *first Location* header’s *host* for the SEND for normal EC write responses.
- Use the *last Location* header’s *host* for non-EC write responses and for SEND after multipart completes.

Request headers – No special headers are expected or used on the request. Do *not* include the legacy SEND request headers:

- Castor-System-Cluster
- Castor-System-Type
- Castor-System-Auth

Query Arguments for SEND

The SEND request needs query arguments for **feedid**, **feedtype**, or both, which is a union of all those provided arguments; SEND reverts to the [legacy behavior](#) if neither is provided.

Argument	Values, Examples	Notes
admin	none	SEND is used by an <code>admin</code> user. This requires the admin query argument and a system user /password sent as a digest.
feedid	<code>all</code> <i>integer</i> <code>feedid=all</code> <code>feedid=1&feedid=3</code>	<p>Specifies one or more <i>specific</i> feeds as the replication destination. Reference existing feed IDs (<code>feedid=1&feedid=3</code>) or use the special value "all" to refer to all feeds, including no feed. Select Cluster > Feeds, and locate the ID column to find the ID number for a feed in the Swarm UI:</p> 
feedtype	<code>all</code> <code>search</code> <code>replication</code> <code>s3backup</code> <code>feedtype=all</code> <code>feedtype=replication &feedtype=search</code>	<p>Specifies one or more <i>types</i> of feeds as the replication destination, from among these values: <code>search</code>, <code>indexing</code>, <code>replication</code>, <code>s3backup</code>. (The values <code>search</code> and <code>indexing</code> are synonymous.) Use the special value "all" to refer to all feed types, including no feed.</p> <p>Swarm returns an HTTP 400 Bad Request error if the feedtype is not a valid value. The SEND operation succeeds if the object being sent does not match the feed definition (because of a domain restriction), but no data is transferred.</p>
timeout	<code>true</code> <i>number of seconds</i> <code>false</code> <code>timeout=true</code> <code>timeout=60</code>	<p>Sets how long to wait for replication to complete; if disabled (<code>false</code>; <i>not recommended</i>), feed processing can go on indefinitely if a feed is blocked.</p> <p>Using <code>timeout=true</code> waits for the Swarm setting <code>scsp.defaultFeedSendTimeout</code> time in seconds, which defaults to 30.</p> <p>Specifying a positive number for the timeout overrides the value in the Swarm setting.</p>

SEND Responses

The request returns information about that request in the body of the response. SEND operates like a HEAD request, with the headers of the response resembling that of a HEAD request.

Chunked encoding

The SEND response is chunked transfer encoded, so the client of the SEND request must be prepared for chunked transfer encoding. The response body may contain additional leading newlines sent incrementally, which keeps the connection open in long requests. The body of the response can be ignored; the trailing headers are repeated at the end of the body to support clients that cannot handle trailing headers, such as curl.

SEND SCSP returns an HTTP 200 OK (request has been completed), even for timeouts.

Response Headers

Check the replication status for the object in these response headers:

- **Feed-<id>-Status** – (The same value as a verbose HEAD/GET request.) The ID refers to the Swarm-assigned ID field in the feed definition.
 - 0 is a success; no writes remain to be completed.
 - 1 is a timeout.
 - Any other positive number is a failure.
- **Feed-<id>-StatusTime** – (The same value as a verbose HEAD/GET request). The HTTP time of the last replication attempt or success. Feed-<id>-StatusTime are blank if Feed-<id>-Status is 1 because the object has not been processed by the feed.

The response is chunked encoded and may include trailing headers:

- The above feed statuses are provided immediately in the HTTP 200 response headers for feeds with a successful status (0) prior to the request. There are no feed status headers if there is no matching feed.
- Any other feed statuses are provided as trailing headers. A newline keep-alive chunk is sent periodically as per **scsp.keepAliveInterval** during processing. At the end of the request, the trailing headers are sent both in the body of the request and as trailing headers. All results are sent at the end of the response, not when processing completes for an individual feed.
- A failure or timeout response is provided with no automatic retries if a feed-to-be-processed is blocked or paused.

Example of SEND

The following request transfers the unnamed object to all clusters that are the targets of replication feeds:

```
curl -i -X SEND --location-trusted
--anyauth -u admin:ourpwdofchoicehere
"http://192.168.1.12:80/97f7149dec6cbc0aa1e9425688158969
?feedtype=replication&timeout=true&admin"
```

See also [SCSP SEND - legacy](#).

SCSP SEND - legacy

New SEND

As of Swarm 11.2, [SCSP SEND](#) has been expanded to support all feed types and to use the new replication (PUSH) method. For backwards compatibility, the following legacy behavior of SEND is still supported. If a SEND request is missing the new required query arguments, then the behavior described here is in force. (v11.2)

The SCSP SEND method applies to both named and unnamed objects. The SEND request allows explicit transmission of a single object from a source cluster to a remote one, such as for keeping two clusters immediately synchronized. This feature is best used with a [replication feed](#), which acts as a catch-up mechanism if the intracluster network is down or the SEND command should fail.

SEND Requests

SEND can only be used by an `admin` user. With a SEND request, Swarm performs the appropriate GET/retrieve in the destination cluster to verify whether the object already exists there.

Note

You issue SEND *directly* on a Swarm node in the source cluster, to transfer a single object to a destination cluster. Do not send the request to a Gateway or other proxy.

You use the following headers with SEND requests:

Castor-System-Cluster	required	The value of the <code>cluster.name</code> setting of the <i>destination</i> cluster.
Castor-System-Target	required	The <code>IP:port</code> of a node or reverse proxy of the <i>destination</i> cluster.
Castor-System-Auth	optional	The <code>username:password</code> for an administrative account on the remote cluster, only if it differs from the source cluster.

Responses to SEND

The request returns information about that request in the body of the response. SEND operates like a HEAD request, with the headers of the response resembling that of a HEAD request.

Important

The SEND response is chunked transfer encoded, so the client of the SEND request must be prepared for chunked transfer encoding. The response body may contain additional leading newlines sent incrementally, which keeps the connection open in long requests.

The most common HTTP response codes:

- **201** - the object has been transferred successfully
- **409** - the object already exists in the destination cluster

Example of SEND

The following request transfers the object to the "dr" cluster at 192.168.1.13, with a 201 in that cluster:

```
curl -i -X SEND --location-trusted
-H "Castor-System-Cluster: dr"
-H "Castor-System-Target: 192.168.1.13:80"
--anyauth -u admin:ourpwdofchoicehere
"http://192.168.1.12:80/97f7149dec6cbc0aa1e9425688158969?alias&admin"

HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest realm="CASTor administrator", nonce="0c6da76911cbf5cd495afbb0c66e6d9a",
opaque="3e894c0cf7clad980elfd46320307f1a", stale=false, qop="auth", algorithm=MD5
WWW-Authenticate: Basic realm="CASTor administrator"
Content-Length: 53
Content-Type: text/html
Date: Thu, 18 Apr 2016 15:13:00 GMT
Server: CASTor Cluster/9.1.0
Allow: HEAD, HOLD, GET, SEND, PUT, RELEASE, POST, COPY, GEN, APPEND, DELETE

HTTP/1.1 200 OK
Castor-System-Alias: 97f7149dec6cbc0aa1e9425688158969
Castor-System-Cluster: baker
Castor-System-Created: Thu, 18 Apr 2016 15:10:40 GMT
Castor-System-Version: 1366297840.592
Content-type: text/xml
Last-Modified: Thu, 18 Apr 2013 15:10:40 GMT
transfer-encoding: chunked
Etag: "166e93908fc32ffb5f55beb7ed531ba1"
Volume: 8f61a5127994365e3dd89bbf83aa0964
Volume-Hint: b79cf7801f71f545c62957ae5659299b
Date: Thu, 18 Apr 2016 15:13:01 GMT
Server: CASTor Cluster/9.1.0

Remote cluster returned: 201
```

Metadata Headers

Header limits – Headers (metadata) are constrained not only by Swarm but by all services, proxies, and clients (such as Elasticsearch, Twisted, Jetty, HAProxy) handling objects. A Swarm object may have any number of [annotations](#) associated with it, but each annotation object is subject to these limits. The persisted metadata on Swarm objects must fall within these limits by default:

- 500 – The *total number* of headers on an object
- 32 KB – The *combined length* of all headers (key/value pairs); exceeding this returns a 400 (Bad Request) error
- 16 KB – The *maximum length* for a given header (key/value pair)
- [Metadata Annotation](#)
- [Custom Metadata Typing](#)
- [Allow Metadata Header](#)
- [Encoding Non-ASCII Characters in Metadata](#)
- [Custom Metadata Headers](#)
- [Lifepoint Metadata Headers](#)

Metadata Annotation

- [Important](#)
- [Note](#)
- [Annotation Cleanup](#)
 - [Note](#)
- [Creating Annotations](#)
 - [Tip](#)
 - [Extending metadata with post-processed data](#)
 - [Alias objects](#)
- [Searching for Annotations](#)
- [Sample Scenario for Annotations](#)
- [Adding Metadata Annotation](#)
 - [Important](#)
 - [Tip](#)
 - [Note](#)

In addition to updating object metadata directly (via [COPY](#)), append additional metadata to existing objects without altering the original. This provides a method to extend the metadata of immutable objects, including historical versions, because each object's create date, original metadata, and version sequence remain undisturbed. Annotations provide an additional method for finding and managing objects, such as storing S3 object-level ACLs for the Gateway to enforce.

 **Important**

Swarm cannot be downgraded to an earlier version once this feature is used.

Benefits - Keeping metadata annotation separate from the object itself provides several advantages:

- Add helpful metadata without changing the object's create date, original metadata, and version sequence.
- Retrieve objects as originally written, so applications can distinguish between what was original and what was added later.
- Annotate *immutable* objects.
- Annotate *historical versions* of objects, independent of the current version of the object. This is keenly important when the metadata is derived from analysis performed on the data, which changes from version to version, or when capturing information about specific versions.

 **Note**

- This lightweight implementation of annotation does not rely on the annotator and the target object interacting, and the objects do not operate as a pair. For example, there is no single request that returns both objects' headers, and there is no method to merge and resolve conflicts between them.
- It is recommended to use [SCSP COPY](#) operation to update the metadata on an object so the metadata is directly available in listings. This COPY operation is efficient even for large objects, assuming they are erasure-coded, because the manifest stream is copied. For S3, use the PUT with [copy](#) operation, specifying the same source and destination.

Annotation Cleanup

There are two key features of this annotation method: (1) *validation* that target objects exist before annotations are written, and (2) the Health Processor's *automated tracking and cleanup* of annotation objects after the target object is removed. A target object annotated may be removed from Swarm in one of several ways:

- SCSP Delete
- SCSP Write (invalidating the old version)
- Lifepoint Delete
- Recursive delete of a parent context (domain or bucket)



Note

The Health Processor it logs a "DECORATION DELETE" AUDIT-level message when purging an annotation during garbage collection. Annotation objects "decorate" a targeted content object.

Regardless of the type of Swarm object annotated (named, alias, immutable, historical version) and the protection type (replicated or erasure-coded), metadata annotation operate largely the same way:

- Swarm deletes the orphaned annotation during garbage collection if an annotation is created and the target object is later deleted.
- The target object is completely unaffected if an annotation is created and later deleted.
 - For named objects only, Swarm replaces the annotation object with a delete marker.
- Swarm deletes both recursively if deleting a domain or a bucket containing both the original object and the annotation.
- Create separate annotations for any historical versions if updating a versioned object; Swarm deletes the orphaned annotation during garbage collection when deleting a version.
- Two outcomes occur based on the position in the version chain if creating and later deleting an annotation on a versioned object:
 - *Historical versions*: Swarm removes the annotation.
 - *Current versions*: Swarm replaces the annotation object with a delete marker.

Creating Annotations

Metadata annotation makes use of a persisted header, `Castor-System-Decorates`, which is the ETag of the target object the annotation object is extending (decorating). This is an annotation object, subject to special Health Processor management, if this header is present. The header is valid for all Swarm object types (immutable, alias, and named), but not for context objects (domains and buckets). Both the annotator (decorator) and annotated target object may be versioned.



Tip

Although it is common to create annotations as metadata-only (`Content-Length: 0`) objects, annotations are complete objects. Data may be included as part of the annotation.

Create a new annotation object create an object pointing to the ETag of the target and includes the custom metadata to be added, such as GPS coordinates extracted from an existing, uploaded photo:

Extending metadata with post-processed data

```
Content-Length: 0
Castor-System-Decorates: 9282727ffcca3a09e0843281aafc13af
X-GPS-Meta-Longitude: 36; 16; 48.360000000000589
X-GPS-Meta-Latitude: 115; 10; 20.79299999981990
```

Note: the ETag has no quotes, even though returned ETags are quoted. The ETag must be of a content-containing object, not a delete marker.

 **Alias objects**

The current ETag of an alias object, not the permanent UUID, must be used for creating annotations. Swarm returns a 400 - Bad Request error if using the UUID.

Searching for Annotations

In the annotation (decorator) object’s Elasticsearch record, the `Castor-System-Decorates` header value is indexed under the key **decorates**, and the Elasticsearch configuration templates include the **decorates** field. Most Swarm queries return this value, if present, as part of the results.

Query argument - Use a “`decorates=<uuid>`” query argument in Swarm listing queries to find annotation objects for a given ETag (or earlier query result “hash”).

See [Listing Operations](#).

Sample Scenario for Annotations

Suppose a company needs to store surveillance videos as immutable objects (as protection from tampering) in the domain “`swarm.example.com`”. To add a video, use the normal POST, adding the Content-Type of the video and custom metadata for the video's duration, camera location, and camera model:

```
curl -i --location-trusted -X POST --post301 \
--data-binary @20170311-972-9928817883.mp4 \
-H "Expect: 100-continue" \
-H "x-example-meta-Start-Time: 2017-03-11T12:00:01.678Z" \
-H "x-example-meta-End-Time: 2017-03-11T13:00:00.421Z" \
-H "x-example-meta-Building: Annex 2" \
-H "x-example-meta-Location: 972" \
-H "x-example-meta-CameraModel: SWDSK-850004A-US" \
-H "Content-Type: video/mp4" \
-H "Content-Disposition: inline" \
"http://swarm.example.com/"
```

```
HTTP/1.1 201 Created
Location: http://192.168.1.11:80/e970b3280d5501571c8c6fe9d6838557?domain=swarm.example.com
Location: http://192.168.1.12:80/e970b3280d5501571c8c6fe9d6838557?domain=swarm.example.com
Volume: b3381183a1cfc620d960db3eae1d086d
Volume: 604a44d1a351045553b5481391af0810
Manifest: ec
Content-UUID: e970b3280d5501571c8c6fe9d6838557
Last-Modified: Tue, 28 Mar 2017 19:19:48 GMT
Castor-System-Encoding: zfec 1.4(2, 1, 524288, 200000000)
Castor-System-Version: 1490728788.934
Etag: "681b2470307b9260fb83542903e51828"
Replica-Count: 2
Date: Tue, 28 Mar 2017 19:22:19 GMT
Server: CASTor Cluster/9.2.0
Content-Length: 46
Content-Type: text/html
Keep-Alive: timeout=14400
<html><body>New stream created</body></html>
```

To verify the video is successfully stored, use a HEAD command:

```
curl --head --location-trusted "http://swarm.example.com/e970b3280d5501571c8c6fe9d6838557"
```

```
HTTP/1.1 200 OK
Castor-System-CID: 7e7fd5d747d244726af93c726672408b
Castor-System-Cluster: swarm.example.com
Castor-System-Created: Tue, 28 Mar 2017 19:19:48 GMT
Content-Disposition: inline
Content-Type: video/mp4
Last-Modified: Tue, 28 Mar 2017 19:19:48 GMT
x-example-meta-Building: Annex 2
x-example-meta-CameraModel: SWDSK-850004A-US
x-example-meta-End-Time: 2017-03-11T13:00:00.421Z
x-example-meta-Location: 972
x-example-meta-Start-Time: 2017-03-11T12:00:01.678Z
Manifest: ec
Content-Length: 1500964975
Etag: "681b2470307b9260fb83542903e51828"
Castor-System-Domain: swarm.example.com
Volume: b3381183a1cfc620d960db3eae1d086d
Date: Tue, 28 Mar 2017 19:24:25 GMT
Server: CASTor Cluster/9.2.0
Keep-Alive: timeout=14400
```

The custom metadata is what makes it possible and practical to identify video of interest. Suppose an incident occurs in the Annex 2 building. Search for immutable video taken at Annex 2 during the time span to find surveillance video relevant to the investigation:

```
curl -i --location-trusted "http://swarm.example.com/?domain=swarm.example.com&format=json&fields
&stype=immutable\
&content-type=video/mp4\
&x-example-meta-Building=Annex%20\
&x-example-meta-Start-Time:date=<2017-03-11T12:17:23Z\
&x-example-meta-End-Time:date=>2017-03-11T12:17:23Z"

HTTP/1.1 200 OK
Castor-System-Alias: 7e7fd5d747d244726af93c726672408b
Castor-System-CID: ffffffff
Castor-System-Cluster: swarm.example.com
Castor-System-Created: Tue, 28 Mar 2017 19:19:29 GMT
Castor-System-Name: swarm.example.com
Castor-System-Owner: @CAStor administrator
Castor-System-Version: 1490728769.536
X-Timestamp: Tue, 28 Mar 2017 19:19:29 GMT
Last-Modified: Tue, 28 Mar 2017 19:26:30 GMT
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8
Castor-Object-Count: 1
Castor-System-Object-Count: 1
Date: Tue, 28 Mar 2017 19:26:30 GMT
Server: CAStor Cluster/9.2.0
Keep-Alive: timeout=14400
[
  {
    "contextid": "7e7fd5d747d244726af93c726672408b",
    "x_example_meta_start_time": "2017-03-11T12:00:01.678Z",
    "x_example_meta_end_time:date": "2017-03-11T13:00:00.421Z",
    "@timestamp": 1490728939869,
    "domainid": "7e7fd5d747d244726af93c726672408b",
    "last_modified": "2017-03-28T19:19:48.932400Z",
    "bytes": 1500964975,
    "hash": "681b2470307b9260fb83542903e51828",
    "x_example_meta_location:double": 972.0,
    "content_disposition": "inline",
    "sizewithreps": 2251447463,
    "content_type": "video/mp4",
    "timestamp": 1490728939869,
    "x_example_meta_location:long": 972,
    "x_example_meta_location:date": 972000,
    "x_example_meta_end_time": "2017-03-11T13:00:00.421Z",
    "name": "e970b3280d5501571c8c6fe9d6838557",
    "castor_stream_type": "immutable",
    "x_example_meta_building": "Annex 2",
    "x_example_meta_location": "972",
    "x_example_meta_cameramodel": "SWDSK-850004A-US",
    "x_example_meta_start_time:date": "2017-03-11T12:00:01.678Z"
  }
]
```

The search correctly finds a video of interest: e970b3280d5501571c8c6fe9d6838557

Adding Metadata Annotation

With the video stored securely, suppose the organization also needs to run an application to perform facial recognition on the video. An application generates data when it is run, including both information on the algorithm/settings and the detailed results. The original video object must remain read-only to serve as evidence, so the derived data and metadata must be stored with a method associating it with the original object without altering it.

The solution is to annotate the video with a decoration object (which can be named or unnamed) to associate the results with the original video.



Important

The `Castor-System-Decorates` header always refers to the ETag of the original video, not the GUID; this is a precaution against the annotation becoming orphaned if the target object is mutable.

```
curl -i -X POST --post301 --location-trusted -d @results \
-H "Castor-System-Decorates: 681b2470307b9260fb83542903e51828" \
-H "x-VideoAnalysis-meta-Algorithm: facial-recognition" \
-H "x-VideoAnalysis-meta-Version: 8.7" \
-H "Content-Type: application/vnd.analysis.facerec" \
"http://swarm.example.com"
```

```
HTTP/1.1 201 Created
Location: http://192.168.1.13:80/0cb2d9e90a3341b10bc9dba27f27259c?domain=swarm.example.com
Location: http://192.168.1.12:80/0cb2d9e90a3341b10bc9dba27f27259c?domain=swarm.example.com
Volume: 6bd38289c2a8fb314caf902d9811fb87
Volume: 604a44d1a351045553b5481391af0810
Manifest: ec
Content-UUID: 0cb2d9e90a3341b10bc9dba27f27259c
Last-Modified: Tue, 28 Mar 2017 20:26:08 GMT
Castor-System-Encoding: zfec 1.4(2, 1, 524288, 200000000)
Castor-System-Version: 1490732768.888
Etag: "867c10c9e6649313a3a5eed2cc76f307"
Replica-Count: 2
Date: Tue, 28 Mar 2017 20:26:12 GMT
Server: CAStor Cluster/9.2.0
Content-Length: 46
Content-Type: text/html
Keep-Alive: timeout=14400
<html><body>New stream created</body></html>
```

Tip

Create annotations as part of the intake process that stores the original objects in Swarm.

To find any annotations producing facial recognition on the original object, search for objects that decorate the video and also qualify the search to look for facial recognition results:

```
curl -i --location-trusted "http://swarm.example.com/?domain=swarm.example.com&format=json&styp=
&decorates=681b2470307b9260fb83542903e51828\
&x_videoanalysis_meta_algorithm=facial%20recognition"
```

```
HTTP/1.1 200 OK
Castor-System-Alias: 7e7fd5d747d244726af93c726672408b
Castor-System-CID: ffffffff
Castor-System-Cluster: swarm.example.com
Castor-System-Created: Tue, 28 Mar 2017 19:19:29 GMT
Castor-System-Name: swarm.example.com
Castor-System-Owner: @CAStor administrator
Castor-System-Version: 1490728769.536
X-Timestamp: Tue, 28 Mar 2017 19:19:29 GMT
Last-Modified: Tue, 28 Mar 2017 20:36:40 GMT
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8
Castor-Object-Count: 1
Castor-System-Object-Count: 1
Date: Tue, 28 Mar 2017 20:36:40 GMT
Server: CAStor Cluster/9.2.0
Keep-Alive: timeout=14400
[
  {
    "sizewithreps":11684987,
    "contextid": "7e7fd5d747d244726af93c726672408b",
    "content_type": "application/vnd.analysis.facerec",
    "name": "0cb2d9e90a3341b10bc9dba27f27259c",
    "castor_stream_type": "immutable",
    "timestamp":1490732772033,
    "@timestamp":1490732772033,
    "domainid": "7e7fd5d747d244726af93c726672408b",
    "decorates": "681b2470307b9260fb83542903e51828",
    "x_videoanalysis_meta_algorithm": "facial-recognition",
    "x_videoanalysis_meta_version:long":9,
    "x_videoanalysis_meta_version:date":8700,
    "x_videoanalysis_meta_version": "8.7",
    "hash": "867c10c9e6649313a3a5eed2cc76f307",
    "last_modified": "2017-03-28T20:26:08.888400Z",
    "x_videoanalysis_meta_version:double":8.7,
    "bytes":7789991
  }
]
```

The search correctly finds an annotation: 0cb2d9e90a3341b10bc9dba2

Note

Elasticsearch is a NoSQL (non-relational) database that does not support joins directly, so queries for a primary object and an annotation object cannot be combined.

Custom Metadata Typing

- [Tip](#)
- [Data Types for Custom Headers](#)
 - [Important](#)
 - [Multiple data types](#)
- [Querying Custom Metadata by Type](#)
- [Upgrading an Existing Index](#)
 - [Important](#)

Swarm indexes custom metadata as strings by default. Swarm types the custom metadata if the data follows standards for numeric, time, or geospatial formats, so query operations can be performed and manage the data by type. Swarm applies typing to [Custom Metadata Headers](#) on all context objects (domains, buckets) and content objects (named, alias, immutable, untenanted). (v9.2)

Tip

Prevent the additional typing by including a non-digit, non-"e" character (such as ~) in the string if string data exists that may match the pattern for a number or date.

Data Types for Custom Headers

On POST and COPY, these are the data types Swarm parses and recognizes in custom headers:

- String (default)
- Numeric (double and long)
- Date ([ISO 8601](#))
- Geo-point (latitude, longitude)

Swarm Search indexes on these types so queries can be run against these types in the custom metadata.

Important

Convert dashes (hyphens) to underscores which is how the headers are indexed for Elasticsearch if querying directly to Elasticsearch rather than through Swarm.

Type	Example Header	Example Search	Notes
String	"x-foo-meta: ASCII string."	&x-foo-meta=ASCII%20string	The default datatype. See the Elasticsearch documentation on String data .
Double	"x-foo-meta: 34567.123"	&x-foo-meta:double=34567.123	See the Elasticsearch documentation on Numeric data .

Long	"x-foo-meta: 93e27"	&x-foo-meta:long=93e27	To prevent overflow, Swarm indexes numeric values only up to 64 bits of precision. See the Elasticsearch documentation on Numeric data .
Geo-point	"x-foo-meta: 11, -33" "x-foo-meta: (11, -33)" "x-foo-meta: 11.22, -33.44"	&x-foo-meta:geo=11,-33	A 2-tuple of two numeric values. The first value is <code>lat</code> (latitude); the second value is <code>lon</code> (longitude). For search operations, evaluations are performed element by element. See the Elasticsearch documentation on Geo-point data .
Date	"x-foo-meta: 1420070400001" (milliseconds) "x-foo-meta: 208704067" (seconds) "x-foo-meta: 2016-12-11" "x-foo-meta: 2016-12-11T12:45:30" "x-foo-meta: 2016-12-11T12:45:30.678Z" "x-foo-meta: 2016-12-11T12:45:30.678-01" "x-foo-meta: 2016-12-11T12:45+01:23"	&x-foo-meta: date=1420070400001	Counts of time since the epoch can be numbers up to 13 digits: <ul style="list-style-type: none"> • Digits between 11 and 13 are read as a date in milliseconds-since-the-epoch • Digits fewer than 11 are read as a date in seconds-since-the-epoch <p>For dates, the time and zone are optional. An offset from UTC (Zulu) can be appended to the time as <code>±[hh]:[mm]</code>, <code>±[hh][mm]</code>, or <code>±[hh]</code>.</p> <p>See the ISO 8601 Standard and the Elasticsearch documentation on Date data.</p>



Multiple data types

Every custom header value is indexed as type String; if its format happens to *also* match one or more of the other data types, Swarm indexes each of those in *addition* to String:

```
{
  ... ,
  "x-foo-meta" : "34684030120",
  "x-foo-meta:long" : 34684030120,
  "x-foo-meta:double" : 34684030120.0,
  "x-foo-meta:date" : 34684030120,
  ...
}
```

Querying Custom Metadata by Type

Query for custom metadata typed by using the additional fields when making SCSP queries to Swarm:

- `&x-foo-meta` performs string-based queries
- `&x-foo-meta:long` performs integer/long typed queries
- `&x-foo-meta:double` performs double typed queries
- `&x-foo-meta:date` performs date-typed queries
- `&x-foo-meta:geo` performs geo-point-typed queries

Here is an example of a query that filters between a range of timestamps, stored in custom headers:

```
curl -i --location-trusted "http://192.168.1.11/surveillance?domain=example.com&format=json&field
&content-type=video/mp4\
&x_example_meta_Building=Annex%202\
&x_example_meta_Start_Time:date=>2017-03-11T12:00Z\
&x_example_meta_End_Time:date=<2017-03-11T12:30Z"
```

Upgrading an Existing Index

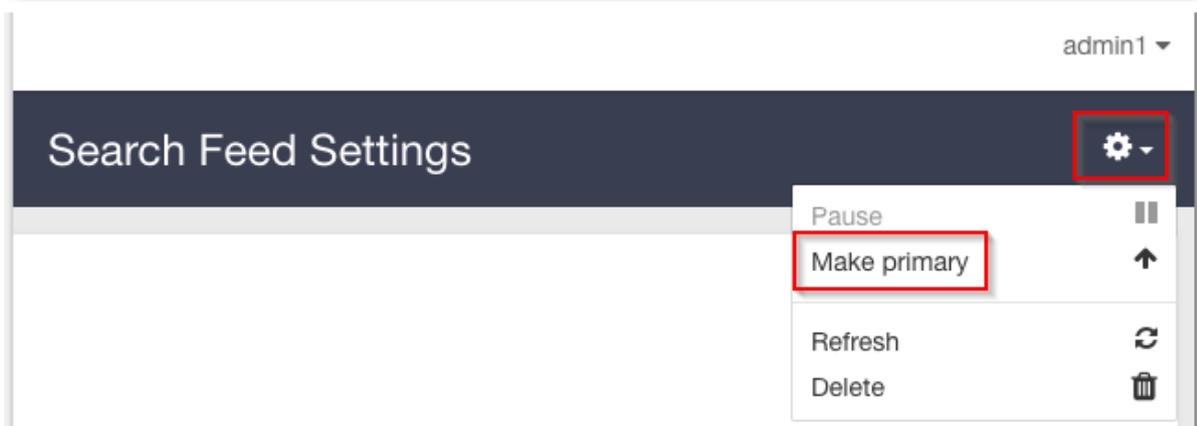
The index needs to be rebuilt if existing custom metadata is currently stored as strings and it is desired to take advantage of typing across all existing custom metadata.

When the underlying schema for Swarm Search changes, new feeds are required to generate index data in the new format. Swarm Storage allows creation of more than one Search feed so transitioning from using one feed to another is possible without disruption. Continue using the primary feed for queries during the transition; the second feed is incomplete until it fully clears its backlog. Transition to it (marking it as primary) as soon as reasonable for your operations when the second feed is caught up.

Important

Delete the original feed when verifying the new primary feed target is working. Having two feeds is for temporary use only because every feed incurs cluster activity, even when paused.

1. [Create a new search feed](#) in the Swarm UI. *Do not select **Make primary**.*
2. Wait until the new feed has completed indexing the cluster, when the feed shows 0 "pending evaluation".
3. Make it the primary feed when the new feed is ready. Navigate to **Cluster > Feeds**, open the new Search feed, and select **Make primary** from the drop-down menu in the Swarm UI.



4. Operate with both feeds for several days. You can restore the old feed to be primary during troubleshooting if there is a problem.
5. Delete the old feed after this confirmation period. Navigate to **Cluster > Feeds**, open the old Search feed, and select **Delete** from the drop-down menu in the Swarm UI.
6. Delete the old index data to reclaim that space if desired.

```
curl -XDELETE 'http://old-elasticsearch:9200/_all' # do not do this to your production data
```


Allow Metadata Header

- [Named objects](#)
- [Disallowing deletes](#)
- [Allow for alias objects](#)
- [Administrative Override](#)
 - [Tip](#)
 - [Evaluating success](#)

The HTTP Allow entity header is used to specify which HTTP methods can be executed for an unnamed object.

ⓘ **Named objects**

Allow headers do not work for named objects; they are used with unnamed objects.

ⓘ **Disallowing deletes**

Allow headers have *no effect* on automatic deletes specified in [Lifepoint headers](#). For best protection from deletes, always use `deletable=no` lifepoints. Using lifepoints allows blocking recursive deletes when a bucket or domain is deleted, causing Swarm to log a CRITICAL error that non-deletable content is present.

See [SCSP Headers](#).

Allow for alias objects

The `GET` and `HEAD` methods are always supported (regardless of the `Allow` header), so there is no need to include an `Allow` header on an unnamed object. For alias objects, the `Allow` header can meet several use cases. Use the following header with a `PUT` request when the user is ready for the object to become immutable if an alias object needs to be mutable for a short time:

```
Allow: GET, HEAD
```

This removes `PUT`, `COPY`, and `APPEND` from the supported methods, effectively *making* the object immutable.

The SCSP server first examines the metadata for the presence of an `Allow` header when asked to perform a request on an alias object. It returns a 405 - Method not allowed response for any method not found in the list if found. The error response includes the `Allow` header stored with the alias object to provide guidance to the application about which methods are allowed for this object. The default result is to allow all methods except `POST` for alias objects if no `Allow` header is stored with the alias object.

An alias object is dynamically deletable if the `DELETE` is included in the `Allow` header and its current lifepoint allows deletes (`deletable=yes`). The `Allow` header has no effect on automatic deletes specified in lifepoint headers, which cause an object to be deleted at a certain point in time. Such lifepoint deletes do not require executing a `DELETE` method, and therefore do not contradict any `Allow` header that may not include support for deletes.

Administrative Override

To prevent content from being stranded with no ability to delete or update it due to an overly restrictive Allow header, Swarm supports administrative override of the Allow header.



Tip

The primary use for the override is to update the Allow header using a COPY request to enable additional needed SCSP methods.

Use the query argument `admin[=yes|true]` with the request to apply the administrative override. `admin` with no optional argument defaults to `true`. Any value other than `yes` or `true` is interpreted as `false` and the administrative override request is ignored.

The `admin` query argument indicates Swarm should evaluate the request for administrative authorization. The user must be in the **CAStor administrator** user list and include credentials with the request in a standard HTTP Authorization header in addition to including the `admin` query argument, , as defined by the HTTP/1.1 spec and the corollary [HTTP Authentication specification](#).

Example of an Authorization header:

```
Authorization: Digest username="JoAdmin",
realm="Castor administrator",
uri="94845f16-c7a8-4606-a62c-6cca639ac358",
response="credentials_digest"
```

- **Bad credentials:** If the administrative request does not include an Authorization header with suitable administrative credentials, Swarm responds to the request with 401 Unauthorized, which includes a WWW-Authenticate challenge containing the administrative domain named `Castor administrator` and other required items.
- **Good credentials:** If the request includes both the query argument and authorized administrator credentials, it proceeds and the Allow header is ignored.

Administrative overrides cannot be used for unsupported methods for an object. Update methods like PUT, COPY, or APPEND to immutable objects. Consider writing the object as an alias object with an Allow header not including the update methods if immutability may need to be overridden in the future: This prevents normal users from modifying the object but allows the administrator to update it using an authorized administrative request, if needed.

All administrative requests are logged along with the user name of the requester for audit purposes.

Evaluating success

Swarm examines the following to determine whether a particular SCSP method succeeds:

1. The `admin` query argument bypasses other authorization methods.
2. The methods allowed by the Allow header.

Encoding Non-ASCII Characters in Metadata

All non-ASCII characters in object headers must be rendered into a standard format for accurate metadata indexing and querying with Elasticsearch. Swarm supports character encoding so non-ASCII characters may be used in the HTTP headers of objects.

- [How Swarm Handles Non-ASCII](#)
 - [Exception](#)
 - [Gateway](#)
 - [Note](#)
- [How to Encode Non-ASCII Characters](#)
 - [Note](#)
 - [Best practice](#)
- [Examples of Decoding](#)
 - [Valid header values](#)
 - [Valid but malformed header values](#)
 - [Invalid header values](#)
- [Decoding Limitations](#)
- [Troubleshooting Decoding](#)
 - [Problems in encoded word structure](#)
 - [Unknown or unreadable encodings](#)
 - [Problems with Base64 encoding](#)
 - [Problems with one of several encoded words](#)

How Swarm Handles Non-ASCII

The presence of non-ASCII characters in object metadata requires extra processing by Swarm, which needs to create a standard format for indexing in Elasticsearch. Swarm can support these encodings with full metadata searching if encoding these characters correctly. Here is a summary of how Swarm handles these characters:

- **Swarm tolerates validation failures.** Swarm tolerates validation failures and stores header values that are left unencoded, so it does not disturb existing objects whose stored headers may fail decoding under the new header rules. Swarm does not reject any object based on an inability to decode encoded words in a header.
- **Swarm stores and returns header fields as-is.** Swarm allows all string-typed headers to have multiple lines as well as encoded words. Swarm stores the header value as-is with the object metadata. When Swarm needs to use that value (such as for metadata indexing) it decodes the value. Swarm decodes header fields into Unicode and then operates on the decoded values. The original encoded persistent headers remain safely stored with the object and are returned performing HEAD or GET operations against the object.

Exception

Different line breaks may display in multiple-line headers, since Swarm does not store the actual line breaks.

- **Metadata goes to Elasticsearch as Unicode.** Swarm sends metadata to Elasticsearch as document attributes through the Elasticsearch API. Swarm decodes the metadata using the algorithms specified in [RFC 2047](#) to achieve this.
- **ISO-8859-1 encoding for headers.** Swarm follows the HTTP/1.1 specification, which defines request header values as ISO-8859-1 characters. Swarm allows header values to be encoded according to [RFC 2047](#) to store *any* Unicode characters (not isolated to ASCII or ISO-8859-1). Applications can encode header values as RFC 2047 'encoded-words' to safeguard treatment of non-ASCII characters. This encoding uses *only* ASCII characters. Swarm decodes it to get the original non-ASCII strings for indexing and searching.

Gateway

With Gateway, use only ASCII (not ISO-8859-1) characters in header values, even though ISO-8859-1 works with Swarm. ASCII header values can be [RFC 2047](#) encoded Unicode characters, which support Elasticsearch indexing and searching.

- **Swarm encodes other character sets.** Per HTTP/1.1 [specifications](#), Swarm headers encode the field content so clients can encode characters in sets other than ISO-8859-1.

Note

Decoding affects performance, but the impact is minimal for fields with no special encodings; there is no performance impacts unless large volumes of non-ASCII metadata is stored in a cluster that enabled full metadata searching.

How to Encode Non-ASCII Characters

Suppose a header string with a non-ASCII character is sent to Swarm, such as `café`. The non-ASCII characters must be escaped if not encoded in ISO-8859-1. One header value can combine partial- and whole-word encoding:

- *Partial-word* encoding: `café=?UTF-8?Q?=C3=A9?='`
- *Whole-word* encoding: `=?UTF-8?Q?café=C3=A9?='`

The string “café red white café brown orange” can be handled both ways:

X-Alt-Meta-Name: =?UTF-8?Q?caf=C3=A9?= red white =?UTF-8?Q?caf=C3=A9_brown_orange?=-



Note

Use ‘=20’ or underscores (_) to encode embedded spaces.



Best practice

Although Swarm does not enforce compliance, the best practice is to comply with the [RFC2047](#) limits:

- 75 characters per encoded word
- 76 characters per each line with an encoded word

Examples of Decoding

This is how Swarm decodes the following header values:

ASCII	"alpha beta gamma"	'alpha beta gamma'
UTF-8	"=?utf-8?q?caf=C3=A9?="	u'caf\xe9'
UTF-8 Base64	"=?utf-8?b?Y2Fmw6k=?="	u'caf\xe9'
Complex UTF-8	"=?utf-8?q?caf=c3=a9?= aaa =?utf-8?q?caf=c3=a9?= bbb =?utf-8?q?caf=c3=a9?="	u'caf\xe9 aaa caf\xe9 bbb caf\xe9'
ISO-8859-1	"=?iso-8859-1?q?caf=E9?="	u'caf\xe9'

Valid header values

"alpha beta gamma"	Pure ASCII
"=?utf-8?q?caf=C3=A9?="	UTF-8
"=?utf-8?b?Y2Fmw6k=?="	UTF-8 Base64 encoded

Valid but malformed header values

While valid, incompletely formatted encodings are not decoded because Swarm does not recognize them as having been encoded. Swarm treats the content as valid content not encoded when the encoding *format* is malformed.

"=?utf-8?q?caf=C3=A9"	UTF-8 without the expected suffix
"=?utf-8?q?caf=C3=A9="	UTF-8 with a partial suffix
"utf-8?q?caf=C3=A9?="	UTF-8 without the expected prefix
"?utf-8?q?caf=C3=A9?="	UTF-8 with a partial prefix

Invalid header values

"=?utf-8?j?caf=C3=A9?="	UTF-8 with an invalid coding indicator (not "Q" or "B")
"=?utf-8?qcaf=C3=A9?="	UTF-8 missing an internal "?" separator character
"=?utf-9?q?caf=C3=A9?="	Invalid or unknown character encoding
"=?utf-8?q?caf=C3=FF?="	Invalid or unknown character
"=?utf-8?b?Y2-mw6k=?="	Base64 with invalid characters
"=?utf-8?b?Y2Fmw6k=?="	Base64 with invalid padding

Decoding Limitations

Swarm does *not* perform the following:

Feature	Swarm Behavior	Workaround	Example
Disable decoding	Swarm has no configuration parameter to disable decoding of headers. Decoding rules are applied to all header values.	Encode the content itself by encoding as ISO8859-1 with the ? and = replaced by octet values if a header value that looks like an encoded string but is meant to be taken literally is needed.	To have the header value passed as-is to Elasticsearch (instead of being decoded), replace the encoding like this: <pre>"=?UTF-8?Q?caf=C3=A9?="</pre> <pre>"=?ISO-8859-1?Q?=3D=3FUTF-8?Q=3Fcaf=3DC3=3DA9=3F=3D?="</pre>
Unicode normalizing	Swarm does not perform Unicode normalization.	Standardize how such words are encoded if various encodings of a word such as "café" need to match.	Valid variants for encoding diacritics in UTF-8: <pre>=?UTF-8?Q?caf=C3=A9?="</pre> <pre>=?UTF-8?Q?caf=65=CC=81?="</pre>
Unicode case folding	Swarm performs no Unicode case folding.	None. For case-insensitive operations, Swarm <i>always</i> converts uppercase to lowercase, including for non-ASCII characters.	In ASCII, uppercasing a character and then lowercasing it always results in the same character. That is not always the case for Unicode escapes.

Troubleshooting Decoding

Review these possible reasons why Swarm found the encoding incomplete or invalid if Swarm does not decode a header as expected:

Problems in encoded word structure

These examples have validation issues in the structure of the encoded-word framework, such as:

- an incorrect starting or ending sequence
- an issue in the “?” separators between the character or Q/B encoding

Swarm passes these types of strings as unencoded text.

Example	Error
'=?utf-8?Q?_brown=20=20and_blue?'	Missing the closing '='. Per RFC 2047, encoded word must start with “=?” and end with “?”.
'=?utf-8Q?_brown=20=20and_blue?='	Missing the “?” between the “utf-8” and “Q” characters.
'=?utf-8?J?_brown=20=20and_blue?='	J encoding is invalid.
'=?utf-8??Q?_brown=20=20and_blue?='	Extra “?” before “Q”.

Unknown or unreadable encodings

When Swarm encounters an encoded word with an unknown encoding or a valid encoding with any other problem, such as an invalid octet, it passes it through as-is:

Example	Error
'=?utf-9?Q?_brown=20=20and_blue?='	utf-9 is not a known encoding.
'=?utf-8?Q?_brown=20=FFand_blue?='	0xFF is not a valid octet in utf-8.

Problems with Base64 encoding

Swarm passes through the original header as is if either validation fails in the Base64 encoding.

- **Characters:** Base64-encoded words include only the characters A-F, a-f, +, and /; all other characters are invalid. If any invalid characters are present, Swarm treats the entire encoded word as invalid.
- **Padding:** Base64 encodings include groups of 4-character sequences. Base64 encodings have trailing padding (with “=”) to maintain the string as a multiple of four characters. Swarm treats any Base64 encodings that lack the trailing padding as invalid.

Problems with one of several encoded words

HTTP header content can contain more than one encoded word, but Swarm does not partially decode headers. If any encoded word in a header is invalid, the entire header is passed through unencoded.

Swarm ignores the incomplete encoding (treating it like a valid non-encoded word) and decodes the complete word if a header includes both complete encoding and incomplete encoding (text that looks like an encoded word missing either the leading “=?” prefix or ending “?” suffix).

Custom Metadata Headers

- [Note](#)
- [Tip](#)
- [Requirements for Custom Names](#)
 - [Elasticsearch and dots](#)
- [Requirements for Custom Values](#)
 - [Important](#)
- [Sample Scenario for Custom Metadata](#)

You can create custom metadata headers as a means to pass data required by your application. Including custom metadata on stored objects increases the usefulness of your content: it provides information that can be indexed by Elasticsearch and used to find, filter, and analyze the content later.

Note

Swarm stores these headers and supplied values without parsing, validation, or modification.

You work with custom metadata through the WRITE, UPDATE, and COPY methods. The COPY method allows updating and adding to the metadata on objects *after* the initial WRITE.

Tip

With COPY requests, you can add the **preserve [query argument](#)** to guarantee any custom metadata existing on the object is carried over to the copy. To overwrite an existing value, include the header name with the new value on the request. (v9.2)

See [SCSP Headers](#).

Requirements for Custom Names

Characters – For best compatibility going forward, Swarm restricts you to these characters in your custom metadata header names (v9.1):

- letters (both cases, although case-insensitive is consistent with [HTTP/1.1 RFC](#))
- numbers
- dash (hyphen)
- underscore

Elasticsearch and dots

Some versions of Elasticsearch (such as 2.3.3) do not allow dots in normal field names. When indexing objects, Swarm converts any dots in custom metadata field names (`x_foo_meta_2016.12`) to underscores (`x_foo_meta_2016_12`). (v9.1)

Formats – Follow one of these two naming formats when custom headers are defined, or they are silently ignored and not persisted to the storage cluster:

- `x-*-meta`
- `x-*-meta*`

```
x-ExampleCorp-meta-color: blue
```

Requirements for Custom Values

To specify more than one value for the same header, list the values on the same line, separated by commas.

```
x-color-meta: blue, green
```



Important

Do *not* reuse the same header with different values.

For metadata values, use 7-bit US-ASCII characters, or else follow [RFC 2047](#) guidelines for alternate character sets.

```
x-xml-meta-data: <size>large</size><color>blue</color><specialorder/>
```

Verify the total length of *all* persisted metadata, keys and values, does not exceed 32 KB. Metadata over 32 KB results in a **400 Bad Request** error response from Swarm.

Sample Scenario for Custom Metadata

Assume a domain of "example.com" with a bucket called "surveillance", created for storing the company's surveillance videos.

To add a video, POST to the bucket, specifying the Content-Type of the video and including custom metadata to document the video's duration, camera location, and camera model:

```
curl -i --location-trusted -X POST --post301 \
--data-binary @20170311-972-9928817883.mp4 \
-H "Expect: 100-continue" \
-H "x-example-meta-Start-Time: 2017-03-11T12:00:01.678Z" \
-H "x-example-meta-End-Time: 2017-03-11T13:00:00.421Z" \
-H "x-example-meta-Building: Annex 2" \
-H "x-example-meta-Location: 972" \
-H "x-example-meta-CameraModel: SWDSK-850004A-US" \
-H "Content-Type: video/mp4" \
-H "Content-Disposition: inline" \
"http://example.com/surveillance/2017/03/22/20170311-972-9928817883.mp4"
```

```
HTTP/1.1 100 Continue
Date: Mon, 27 Mar 2017 17:15:26 GMT
Server: CASTor Cluster/9.2.0
Content-Length: 0
```

```
HTTP/1.1 201 Created
Location: http://192.168.1.12:80/surveillance/2017/03/22/20170311-972-9928817883.mp4 \
?domain=example.com
Location: http://192.168.1.13:80/surveillance/2017/03/22/20170311-972-9928817883.mp4 \
?domain=example.com
Volume: 8aff01dbe86d6fff1f27b5872bfc8e840
Volume: cef223aalbfc13e356203fdede8489e4
Manifest: ec
Last-Modified: Mon, 27 Mar 2017 17:15:25 GMT
Castor-System-Encoding: zfec 1.4(2, 1, 524288, 200000000)
Castor-System-Version: 1490634925.750
Etag: "c04b7eac90a3f22292581080c32fdd07"
Replica-Count: 2
Date: Mon, 27 Mar 2017 17:17:16 GMT
Server: CASTor Cluster/9.2.0
Content-Length: 46
Content-Type: text/html
Keep-Alive: timeout=14400
<html><body>New stream created</body></html>
```

Use a HEAD command to verify the video is successfully stored:

```
curl --head \
  --location-trusted "http://example.com/surveillance/2017/03/22/20170311-972-9928817883.mp4"
```

```
HTTP/1.1 301 Moved Permanently
Date: Mon, 27 Mar 2017 17:22:50 GMT
Server: CASTor Cluster/9.2.0
Location: http://192.168.1.12:80/surveillance/2017/03/22/20170311-972-9928817883.mp4
?domain=example.com&auth=2db96e4590e029966aecfd0dd96da7e9
Content-Length: 0
Keep-Alive: timeout=14400
```

```
HTTP/1.1 200 OK
Castor-System-CID: fd20ce977b35d0509205b27977d697d3
Castor-System-Cluster: example.com
Castor-System-Created: Mon, 27 Mar 2017 17:15:25 GMT
Castor-System-Name: 2017/03/22/20170311-972-9928817883.mp4
Castor-System-Version: 1490634925.750
Content-Disposition: inline
Content-Type: video/mp4
Last-Modified: Mon, 27 Mar 2017 17:15:25 GMT
x-example-meta-Building: Annex 2
x-example-meta-CameraModel: SWDSK-850004A-US
x-example-meta-End-Time: 2017-03-11T13:00:00.421Z
x-example-meta-Location: 972
x-example-meta-Start-Time: 2017-03-11T12:00:01.678Z
Manifest: ec
Content-Length: 1500964975
Etag: "c04b7eac90a3f22292581080c32fdd07"
Castor-System-Path: /example.com/surveillance/2017/03/22/20170311-972-9928817883.mp4
Castor-System-Domain: example.com
Volume: 8aff01dbe86d6ff1f27b5872bfc8e840
Date: Mon, 27 Mar 2017 17:22:50 GMT
Server: CASTor Cluster/9.2.0
Keep-Alive: timeout=14400
```

The custom metadata is what makes it possible and practical to find videos. Suppose that an incident occurred in the Annex 2 building; to find surveillance video that may be relevant to the investigation, search the **surveillance** bucket for video taken at Annex 2 during that time span:

```
curl -i --location-trusted "http://192.168.1.11/surveillance\
?domain=example.com\
&format=json&fields=all\
&content-type=video/mp4\
&x-example-meta-Building=Annex%202\
&x-example-meta-Start-Time:date=<2017-03-11T12:17:23Z\
&x-example-meta-End-Time:date=>2017-03-11T12:17:23Z"
```

```
HTTP/1.1 200 OK
Castor-System-Alias: fd20ce977b35d0509205b27977d697d3
Castor-System-CID: 72203a85b0f9d7a64a7625c114f8a886
Castor-System-Cluster: example.com
Castor-System-Created: Mon, 27 Mar 2017 16:37:38 GMT
Castor-System-Name: surveillance
Castor-System-Version: 1490632658.361
X-Timestamp: Mon, 27 Mar 2017 16:37:38 GMT
Last-Modified: Mon, 27 Mar 2017 17:26:00 GMT
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8
Castor-Object-Count: 1
Castor-System-Object-Count: 1
Date: Mon, 27 Mar 2017 17:26:00 GMT
Server: CASTor Cluster/9.2.0
Keep-Alive: timeout=14400
[ {
  "sizewithreps": 2251447463,
  "contextid": "fd20ce977b35d0509205b27977d697d3",
  "content_type": "video/mp4",
  "name": "2017/03/22/20170311-972-9928817883.mp4",
  "x_example_meta_end_time:date": "2017-03-11T13:00:00.421Z",
  "@timestamp": 1490635036512,
  "x_example_meta_building": "Annex 2",
  "x_example_meta_location:date": 972000,
  "x_example_meta_location": "972",
  "x_example_meta_cameramodel": "SWDSK-850004A-US",
  "domainid": "72203a85b0f9d7a64a7625c114f8a886",
  "x_example_meta_start_time:date": "2017-03-11T12:00:01.678Z",
  "hash": "c04b7eac90a3f22292581080c32fdd07",
  "timestamp": 1490635036512,
  "x_example_meta_location:double": 972,
  "last_modified": "2017-03-27T17:15:25.748400Z",
  "bytes": 1500964975,
  "content_disposition": "inline",
  "x_example_meta_location:long": 972,
  "x_example_me ta_end_time": "2017-03-11T13:00:00.421Z",
  "x_example_meta_start_time": "2017-03-11T12:00:01.678Z"
}]
```

The search correctly found a video of interest in the **surveillance** bucket and returned the object: 2017/03/22/20170311-972-9928817883.mp4

Lifepoint Metadata Headers

- [Understanding Storage Policies](#)
 - [Lifepoints to prevent deletion](#)
 - [Best practice](#)
 - [Lifecycle evaluation example](#)
 - [Complete lifecycle policy](#)
 - [Note](#)
- [Specifying Lifepoints and Lifecycles](#)
 - [Guidelines for lifepoints](#)
 - [Correct lifepoint](#)
 - [Incorrect delete constraint](#)
 - [Converting chunked to replication](#)
- [Constraints for Replication and Deletion](#)
 - [ReplicationConstraintSpecialist](#)
 - [Important](#)
 - [DeletionConstraintSpecialist](#)
 - [Incorrect delete constraint](#)
 - [Correct delete constraint](#)
 - [Important](#)

Use optional lifepoint headers to define object-specific Swarm replication and retention policies, with varying complexity as the situation requires.

See [SCSP Headers](#).

Understanding Storage Policies

Each node in a storage cluster includes a Health Processor that continuously cycles through the list of content objects it stores on disk to determine what is considered "healthy" for each object at this particular point in the lifecycle. The Health Processor may determine an object needs to have at least three replicas of itself stored within Swarm. This requirement referred to as a *content constraint* or a *constraint* enables the Health Processor to take the appropriate action when needed to verify disk-level and lifecycle data protection.

Specify a constraint when first storing the object in the storage cluster. For mutable or named objects, the constraint can be changed with a COPY or a PUT.

Constraints can also be grouped together and provided an expiration date. This type of constraint group is called a *lifepoint* because it represents a point where the health requirements of an object change. A sequence of lifepoints are collectively called a *storage policy* or a *content lifecycle*.

Lifepoints to prevent deletion

An important use of lifepoints is to protect objects from deletion. Deleting a bucket containing such protected objects generates errors and orphans those named objects.



Best practice

Make the bucket object indelible if maintaining a bucket for indelible objects.

See "DELETE for domains and buckets" in [SCSP DELETE](#).

Lifecycle evaluation example

Assume an object was written to Swarm on June 12, 2015. In the first six months since creation, the object must have at least three replicas and cannot be deleted by any user. In the second six months since creation, the object needs two replicas, and client applications can delete the object. After a year, the object is deleted.

Complete lifecycle policy

```
Lifepoint: [Wed, 12 Dec 2015 15:59:02 GMT] reps=3, deletable=no
Lifepoint: [Sun, 08 Jun 2016 15:59:02 GMT] reps=2, deletable=yes
Lifepoint: [] delete
```



Note

There is one *instance* of the object if there is one *replica* of an object in a cluster. *Replica* and *instance* are synonymous in this context.

Each time the Health Processor (HP) examines the object, it checks the current date to see how to apply the lifepoint policies:

Time frame	Lifepoint Effects	Notes
------------	-------------------	-------

<p>Before the first lifepoint date</p>	<p>Swarm refuses SCSP DELETE requests.</p> <p>HP maintains at least three replicas of the object in the cluster.</p>	
<p>Between the first and second lifepoint dates</p>	<p>Swarm accepts SCSP DELETE requests.</p> <p>HP allows the number of replicas in the cluster to decrease.</p>	<p>Now the lifepoint specifies the <i>deletable</i> constraint enables a client to delete the content by sending an SCSP DELETE message with the object's name or UUID</p>
<p>After the second lifepoint date</p>	<p>Swarm accepts SCSP DELETE requests.</p> <p>HP deletes the object at the first checkup.</p>	<p>The last lifepoint with no end date is in effect <i>indefinitely</i> once it comes in range.</p>

Specifying Lifepoints and Lifecycles

Use a syntax to specify a complete object lifecycle, and specify one or more lifepoints. Attach lifepoint entity headers to an SCSP WRITE message.

The entity header is shown below in Augmented Backus-Naur Form (BNF) syntax:

```
lifepoint = "lifepoint" ":" end-date 1#constraint end-date = "[" [HTTP-date] "]"
constraint = replication-constraint | delete-constraint | deletable-constraint replication-constraint
"reps" ["=" (1*DIGIT | 1*DIGIT:1*DIGIT)] delete-constraint = "delete" ["=" ("yes" | "no")]
deletable-constraint = "deletable" ["=" ("yes" | "no")]
```

Guidelines for lifepoints

Follow these guidelines when creating a lifepoint:

Guideline	Explanation
Make every lifepoint stand alone	<p>Lifepoints do <i>not</i> build upon one another: they stand alone as a complete specification of the constraints that apply to the object in a provided date range. Include the complete set of constraints for a provided end date in the lifepoint header.</p> <p>Correct lifepoint</p> <pre>Lifepoint: [] reps=1,deletable=no</pre>
Provide time in GMT	<p>For HTTP-date, adhere to the Full Date Section 3.3.1 of the HTTP/1.1 specification. The indicated time must be specified in Greenwich Mean Time (GMT). <i>GMT is exactly equal to UTC</i> (Coordinated Universal Time) when dealing with Swarm.</p>
Do not use deletable= without reps=	<p>The delete constraint does not store a value and cannot include end-date :</p> <p>Incorrect delete constraint</p> <pre>Lifepoint: [] reps=1 Lifepoint: [] deletable=no</pre>
Do not delete contexts by lifepoint	<p>Swarm does not allow lifepoint-triggered deletes of contexts (domains and bucket objects) to protect content objects from being orphaned.</p> <p>See SCSP DELETE for guidance on deleting domains and buckets.</p>
Do not replicate chunked uploads	<p>Chunked uploads are erasure-coded automatically, so a request fails if it is chunked <i>and</i> the current lifepoint specifies replication.</p> <p>Specify <i>two</i> lifepoints to convert a chunked upload: have the first specify an EC encoding expiring in one day, and have the second specify the number of replicas going forward:</p> <p>Converting chunked to replication</p> <pre>Transfer-Encoding: chunked Lifepoint: [Wed, 12 Dec 2016 15:59:02 GMT] reps=5:2 Lifepoint: [] reps=3</pre>

**Do not expect
Swarm to
validate
lifepoints**

Swarm does not validate lifepoints when they are added to the cluster to maximize performance. Swarm accepts an invalid lifepoint and later logs an error if the HP cannot parse the lifepoint.

Constraints for Replication and Deletion

Constraint names and values are parsed by Swarm object classes called **ConstraintSpecialists** that maintain one or more related constraints. The **reps** constraint is parsed and maintained by the **ReplicationConstraintSpecialist**. Constraint names are case-sensitive, and constraint names not recognized by the ConstraintSpecialists are ignored. The set of allowable constraints is extensible, and new constraint types may be added to the system in future releases.

Constraint names and arguments recognized by the ConstraintSpecialists in Swarm include:

- **ReplicationConstraintSpecialist**
- **DeletionConstraintSpecialist**

ReplicationConstraintSpecialist

The ReplicationConstraintSpecialist maintains the desired level of redundancy of content objects and verifies they are stored in the most efficient manner. It understands one constraint name: **reps**, which is set by protection type:

- **Replicas** – a single integer value
- **EC** – a tuple of **k:p** integers (such as 5 : 2)

The ReplicationConstraintSpecialist does this by verifying the actual number of replicas or segments for an object is equal to **reps** at all times. A default value is supplied from the node or cluster configuration if a replication constraint is missing from the lifepoint. Cluster administrators have control over some aspects of replication behaviors through [Swarm configuration parameters](#):

- **Replicas** – Place limits on the number of replicas that can be specified by defining **policy.replicas min** and **max**.
- **EC** – Specify the **ec.minParity** to verify all objects have a minimum number of parity segments included for protection. Invalid or conflicting values of the **reps** constraint are ignored, defaults are used, and warnings are written to the log if found in a lifepoint. Lifepoints with erasure coding define what EC level to apply. For example: **lifepoint = [] reps=5:2** expresses an erasure-coded level of 5 data segments and 2 parity segments.

Supported conversion methods

As of v6.5, a storage policy with multiple lifepoints including the following conversion methods are supported:

- Replication to EC
- EC to replication
- One EC encoding to a different encoding



Important

The object size value must be greater than the **policy.ecMinStreamSize** setting, regardless of the specified lifepoint. Otherwise, the object is not erasure-coded and is instead protected with p+1 replicas.

DeletionConstraintSpecialist

The DeletionConstraintSpecialist completely removes a content object at a certain point in time and allows or disallows client applications to delete the content object using the SCSP DELETE request.

DeletionConstraintSpecialist understands two constraint names: **deletable** and **delete**.

- The **deletable** constraint is set to `yes|true` or `no|false`:
 - `yes|true` (default) indicates the object is deletable by any client knowing the name or UUID. The DELETE method must be included in the **Allow** header for a client delete to be allowed.
 - `no|false` prevents any agent from deleting the object during the effective period of the lifepoint. Any attempt to delete the object result in a 403 (Forbidden) response.
- The **delete** constraint does not accept a value. This constraint causes DeletionConstraintSpecialist to delete the content object from the cluster. The result is the same as if a client application had deleted the object.

To avoid ambiguity, when **delete** is present in a lifepoint specification, it must be the sole constraint in that lifepoint because other conditions on a deleted object may not be applicable. Additionally, a delete lifepoint must be specified with an empty end date.

Incorrect delete constraint

```
Lifepoint: [Wed, 08 Jun 2012 15:59:02 GMT] reps=3, deletable=no, delete
```

Correct delete constraint

```
Lifepoint: [Fri, 12 Dec 2011 15:59:02 GMT] reps=3, deletable=no
Lifepoint: [] delete
```



Important

Do not use **deletable=no** and **delete** in the same lifepoint.

Content Integrity

Content integrity refers to the accuracy and consistency (validity) of content over its lifecycle in Swarm storage. Integrity can be lost at various levels:

- Human error or tampering
- Transfer errors, including unintended alterations or data compromise going to or from storage
- Cyber threats (bugs, viruses/malware, hacking)
- Compromised hardware, such as a device or disk crash
- Physical compromise to devices

Best practices for content integrity include multiple approaches:

- Input validation, to preclude the entering of invalid data
- Error detection/data validation, to identify errors in data transmission
- Security measures, such as access control and data encryption

- [Content Integrity Assurance](#)
- [Content-MD5 Checksums](#)
- [Caching Metadata Headers](#)

Content Integrity Assurance

- [Integrity seals](#)
 - [Direct to Swarm](#)
 - [Example of a hashtype request](#)
 - [Example of a complete integrity seal embedded in a Location header](#)
- [Validating reads](#)
 - [Example of validation with read](#)
 - [Important](#)
- [Application-initiated hash upgrading](#)
 - [Important](#)
 - [Example of hash upgrading](#)

Swarm provides methods for allowing applications to obtain and validate integrity guarantees on the stored data. **Integrity** is an independently verifiable guarantee the data returned for a given name or UUID is exactly the same data stored using that name or UUID, perhaps many months or years in the past. This is performed by **hashing** the data using a cryptographic hash algorithm.

Content metadata is *not* included in the hash. If the application stores the name or UUID and the associated hash value, these can be used later to verify the content has not changed, either through accidental or malicious means.

Integrity seals

An integrity seal is a URL containing the object name or UUID, the hash value, and the type of hash algorithm used for the computations.



Direct to Swarm

Integrity Seal upgrades cannot be performed through Content Gateway. Request them directly from the back-end Swarm cluster.

An application can request an integrity seal when it performs a WRITE by including a `hashtype` query string.

Example of a hashtype request

```
POST http://company.cluster.com/?hashtype=md5 HTTP/1.1
```

These are the current allowable hash types:

- md5
- sha1
- sha256
- sha384
- sha512

Swarm replies with a 201 (Created) response that includes a location header with a URL that can later be used to retrieve the data after creating the object and assigning a name or UUID.

In addition to the host and name or UUID, the URL includes the hash type and value computed from the content object. This URL, including the triple name or UUID, hash type, and hash, is known as the content object integrity seal.

Example of a complete integrity seal embedded in a Location header

```
Location: http://129.69.251.143/41A140B5271DC8D22FF8D027176A0821
?hashtype=md5
&hash=7A25E6067904EAC8002498CF1AE33023
```

Validating reads

An integrity seal can be used in a subsequent READ request to validate the data stored in a storage cluster (*any* cluster). By supplying the URL returned in the Location header from the WRITE request (perhaps replacing the host address if connected to a different cluster or node), the application can ask Swarm to validate while reading the data.

Example of validation with read

```
GET http://129.69.251.143/41A140B5271DC8D22FF8D027176A0821
?hashtype=md5
&hash=7A25E6067904EAC8002498CF1AE33023 HTTP/1.1
```

When Swarm receives such a READ request, it recomputes the hash of the stored content using the supplied hash type and compares the computed hash with the hash value in the integrity seal.

- **Match** - The hashes match if the content was not modified or corrupted. Swarm returns the object with the computed digest as a trailing `Location` header.
- **No match** - Swarm drops the connection before sending the object content at the end of the request if the two values do not match.

Because the hash algorithms are published and well-known, users and third parties can independently validate an object stored by Swarm by reading the contents, computing the hash value, and comparing it with the hash value in the seal. By publishing an integrity seal when it is created, it can be verified the stored content is not modified and it has always been associated with the same UUID.



Important

Range headers are not compatible with integrity seals. The connection may be closed prematurely if the seal is incorrect.

Application-initiated hash upgrading

Occasionally, cryptographers and mathematicians may defeat a cryptographic algorithm, making it possible for hackers to generate different content that has exactly the same hash value as previously-stored content. This issue occurred with the md5 and sha1 algorithms, but not the sha256, sha384, or sha512 algorithms.

Unlike other fixed content storage solutions, Swarm allows a user or application to upgrade a hash algorithm for an existing individual integrity seal. This is performed by issuing a READ request with the name or UUID, the current hash type and hash, and specifying a different, presumably stronger, hash type in the *newhashtype* query parameter.



Important

Upgrade the hash promptly before any exploit of the old algorithm becomes well known and available.

Example of hash upgrading

```
GET http://129.69.251.143/41A140B5271DC8D22FF8D027176A0821
?hashtype=md5
&hash=7A25E6067904EAC8002498CF1AE33023
&newhashtype=sha256 HTTP/1.1
```

This READ request first validates the given integrity seal, then reseal it by *wrapping* the content in the new, upgraded hash algorithm – sha256 in the example. Swarm sends a 200 OK response but drops the connection prior to sending the object content if the requested object fails to validate against the integrity seal. A new integrity seal is returned with the new hash type and hash value if the object validates properly.

Content-MD5 Checksums

- [Client-Provided Content-MD5](#)
- [Swarm-Provided Content-MD5](#)
 - [Tip](#)
 - [Validation failures](#)
- [Storing Content-MD5 Headers](#)
- [Content-MD5 and Replication](#)
- [Content-MD5 and Erasure-Coding](#)

Content-MD5 checksums provide an end-to-end message integrity check of the content (excluding metadata) as it is sent to and returned from Swarm. A proxy or client can check the Content-MD5 header to detect modifications to the entity-body while in transit. A client can provide this header to indicate Swarm should compute and check it as it is storing or returning the object data.

See [SCSP Headers](#).

Client-Provided Content-MD5

During a POST or PUT, the client can provide the following Content-MD5 header as specified in [section 14.15](#) of the HTTP/1.1 RFC:

```
Content-MD5 = "Content-MD5" ":" md5-digest
```

Where **md5-digest** is the base64 of the 128-bit MD5 digest (see [RFC 1864](#) for more information).

The md5-digest is computed based on the content of the entity body, including any content coding applied, but not including any transfer-encoding applied to the message body.

- If this header is present, Swarm computes an MD5 digest during data transfer and then compares the computed digest to the digest provided in the header.
- When completed, the Content-MD5 data is stored with the object and returned with the GET or HEAD request.
- If the hashes do not match, Swarm returns a **400 Bad Request** error response, abandons the object, and closes the client connection.

Swarm-Provided Content-MD5

Another way to associate a Content-MD5 value with an object is to have Swarm compute the ContentMD5 for the body data of the request. Include the **gencontentmd5** [query argument](#) in the request to perform this. Swarm returns the Content-MD5 as a header in the 201 Created response. Once computed, the Content-MD5 data is stored with the object and returned as a response header for any subsequent GET or HEAD requests. Note: the gencontentmd5 query argument replaces use of the "Expect: Content-MD5" request header, which is deprecated per [RFC 2731](#). (v9.2)



Tip

The Swarm setting [scsp.autoContentMD5Computation](#) automates Content-MD5 hashing. The gencontentmd5 query argument or the deprecated Expect: Content-MD5 header on writes does not need to be included (although a separate Content-MD5 header may want to be supplied for content integrity checking). This setting is ignored wherever it is invalid, such as on a multipart initiate/complete or an EC APPEND. (v9.1)

Ranges - When including ?gencontentmd5 on a GET request with a Range header, any Content-MD5 header stored with the object is omitted in the response headers. Instead, a Content-MD5 of the selected range is returned as a trailing header to the GET request.

For details about Range headers, see [section 14.35 \(Range\)](#) in the [HTTP/1.1 RFC](#).



Validation failures

Because of the way Swarm reports a hash validation failure, SCSP reading operations that request a Content-MD5 hash validation and for which there is a hash mismatch causes a storage node to be removed for the Gateway's connection pool temporarily.

Storing Content-MD5 Headers

Content-MD5 headers are stored with the object metadata and returned on all subsequent GET or HEAD requests.

- If a Content-MD5 header is included with a GET request, Swarm computes the hash as the bytes are read, regardless of whether the header was originally stored with the object
- If the computed and provided hashes do not match, the connection is closed before the last bytes are transmitted, which is the standard way to indicate something went wrong with the transfer.

Content-MD5 and Replication

When providing the `gencontentmd5` query argument in a request on a replicated object, the following applies:

- On a write request (POST, PUT, COPY, or APPEND), the Content-MD5 is calculated, stored with the object, and returned as a response header for that write operation.
- The Content-MD5 is always returned for any GET or HEAD request written with the `gencontentmd5` query argument.
- When including `?gencontentmd5` on a range read (a GET request with the Range header), Swarm suppresses any stored Content-MD5 from the response headers and instead return a Content-MD5 for the requested range as a trailing header.

Content-MD5 and Erasure-Coding

When providing the `gencontentmd5` query argument in request on an erasure-coded object, the following applies:

- The APPEND operation is no longer supported. If providing a `gencontentmd5` query argument on an APPEND, it returns a 400 Bad Request error response.
- The COPY operation is supported if providing a `gencontentmd5` query argument on the existing object's write; otherwise the COPY operation fails.
- For a range read (a GET request with the Range header), Swarm suppresses any stored Content-MD5 from the response headers and instead return a Content-MD5 for the requested range as a trailing header.

Caching Metadata Headers

- [Note](#)
- [HTTP 1.1 Caching Headers](#)
 - [Cache-Control](#)
 - [ETag](#)
 - [Note](#)
 - [If-Match](#)
 - [If-None-Match](#)
 - [If-Range](#)
- [HTTP 1.0 Caching Headers](#)
 - [Warning](#)
 - [Last-Modified](#)
 - [Castor-System-Created deprecated](#)
 - [If-Modified-Since](#)
 - [Note](#)
 - [Best practice](#)
 - [If-Unmodified-Since](#)
 - [Expires](#)

Caching metadata headers allows clients and caching proxies quickly determine if a resource was modified since the last time it was read. With alias objects, caching headers allow clients to verify the previous read is the *current* revision before writing an update to it.

See [SCSP Headers](#).

HTTP defines several header mechanisms for clients and caching proxies to quickly determine whether a resource was modified since the last time the data was read. In the Swarm context, caching headers make proxies more effective by extending the caching period to its maximum value, essentially telling the proxies that the resource does change for immutable objects.

To maintain compatibility with a wide variety of browsers and proxies, Swarm implements the caching mechanisms for both HTTP/1.0 and HTTP/1.1.



Note

Swarm returns an HTTP 412 response if it cannot find the bucket or domain associated with a request. This can be distinguished from a cache response by the lack of the current **ETag** in the response headers and a response body that denotes that the bucket or domain cannot be located.

HTTP 1.1 Caching Headers

The newer HTTP/1.1 cache coherency mechanism does not use dates or timestamps and thus avoids the granularity and synchronization problems of the HTTP/1.0 headers. Instead, it uses entity tags (or **ETags**) that can be compared for exact equality.

In Swarm, ETag values are opaque, variable length, case-sensitive strings enclosed in quotes. Any characters preceding or following the quoted string are ignored. If the header value has no quoted string, the entire header is ignored. The value of each date header adheres to the Full Date specification ([RFC 7232](#)), and dates in that format are recognized by Swarm on incoming requests.

Swarm supports the following HTTP 1.1 caching headers:

- Cache-Control ([RFC 7234 5.2](#))
- ETag ([RFC 7231 2.3](#))
- If-Match ([RFC 7232 3.1](#))
- If-None-Match ([RFC 7232 3.2](#))
- If-Range ([RFC 7233 3.2](#))

Cache-Control

Cache-Control can be used on **READ** and **WRITE** requests to determine whether data retrieved from the content cache is acceptable for this request or whether a specific object can ever be stored in the content cache. Swarm supports the **Cache-Control: no-cache** and **Cache-Control: max-age** parameters as discussed in [RFC 7234 5.2](#).

Swarm also supports the **Cache-Control: no-cache-context** extension that instructs Swarm not to use cached contexts. (A **context** is a container; for example, the context of a named object is a bucket.) **Cache-Control: no-cache-context** can be used on any SCSP **READ** or **WRITE** request to instruct Swarm to ignore the content cache when looks up the bucket and domain for a named object. Use it in a **READ** request to prevent Swarm from returning "stale" bucket and domain data from the cache.

See [Use the Content Cache in a Distributed System](#) for when Swarm may return "stale" data.

ETag

Swarm returns the ETag header for all POST, PUT, COPY, APPEND, GET, and HEAD operations. Swarm uses "strong" ETags (as defined in [RFC 7232 2.3](#)) that can be compared for exact (case-sensitive) equality.

Example of an ETag response header:

```
ETag: "508941dc9b52243f64d964b058354b76"
```

The ETag of an immutable unnamed object does not change during the entire lifecycle of the object, whereas mutable named and unnamed object ETags change each time the object is mutated by a **PUT**.



Note

SCSP operations (Update, Delete, etc.) cannot be performed for an existing object using the ETag.

If-Match

A Swarm client or proxy can include the **If-Match** header with the **PUT**, **COPY**, **APPEND**, **GET**, and **HEAD** methods. The value of the header is either a single quoted string (possibly with some ignored flags outside the quotation marks), a comma-separated list of quoted strings, or a single asterisk. Any additional strings are ignored.

Below are examples of **If-Match** request headers:

```
If-Match: "508941dc9b52243f64d964b058354b76"
If-Match: "508941dc9b52243f64d964b058354b76", "fe3233d3c6881d5e8b654117b829d26c"
If-Match: W/ "508941dc9b52243f64d964b058354b76"
If-Match: *
```

Swarm performs the requested method as if the **If-Match** header field did not exist if the entity tags match the primary UUID of the object returned in the response to a similar **GET** request (without the **If-Match** header) on that resource or if "*" is given. The **If-Match** header is ignored if the request results in anything other than a **2xx** status without the **If-Match** header field.

Swarm does not perform the requested method, and instead return an **HTTP 412 Precondition Failed** response with a current ETag header if none of the entity tags match. This behavior is most useful when the client wants to prevent an updating method (such as **PUT**) from modifying an aliased object that changed since the client last retrieved it.

Swarm does **not** return a response status of **HTTP 412 Precondition Failed** unless it is consistent with all conditional header fields in the request if Swarm receives a conditional request that includes both a Last-Modified date (for example, in an **If-Modified-Since** or **If-Unmodified-Since** header field) and one or more entity tags as cache validators (for example, in an **If-Match** header field).

If-None-Match

A Swarm client or proxy can include this header with the **PUT**, **COPY**, **APPEND**, **GET**, and **HEAD** methods to make it conditional. This feature allows efficient cached information updates with a minimum amount of transaction overhead. The header value is either a single quoted string (possibly with some ignored flags outside the quotation marks), a comma-separated list of quoted strings, or a single asterisk, anything after which is ignored.

Examples of **If-None-Match** request headers:

```
If-None-Match: "508941dc9b52243f64d964b058354b76"
If-None-Match: "508941dc9b52243f64d964b058354b76", "fe3233d3c6881d5e8b654117b829d26c"
If-None-Match: W/ "508941dc9b52243f64d964b058354b76"
If-None-Match: */
```

Swarm does not perform the requested method if the entity tags match the primary object UUID that is returned in the response to a similar **GET** request (without the **If-None-Match** header) on that object or if "*" is given and the object does exist. Swarm responds with an **HTTP 304 Not Modified** response, including a current ETag header for the object if the request method was **GET** or **HEAD**. Swarm responds with a response of **HTTP 412 Precondition failed** with the same current ETag as the **GET** or **HEAD** response for all other request methods. The object was modified if none of the previously recorded and supplied entity tags match. The requested method proceeds as if the **If-None-Match** header field did not exist.

Swarm does **not** return a response status of **HTTP 304 Not Modified** or **HTTP 412 Precondition failed** unless it is consistent with all conditional header fields in the request if Swarm receives a conditional request that includes both a **Last-Modified** date (for example, in an **If-Modified-Since** or **If-Unmodified-Since** header field) and one or more entity tags (for example, in an **If-None-Match** header field) as cache validators.

If-Range

A Swarm client or proxy can include the `If-Range` header with a `GET` request method to obtain an additional specified portion of the object if it has not changed or the entire object if it has changed. The value of the header can be either a single quoted string (possibly with some ignored flags outside the quotation marks) or an HTTP-date string (unquoted).

Examples of `If-Range` request headers:

```
If-Range: "508941dc9b52243f64d964b058354b76"
If-Range: W/"508941dc9b52243f64d964b058354b76"
If-Range: Tue, 07 Jul 2009 16:25:24 GMT
```

If a client has a partial copy of an object in its cache and wishes to have an up-to-date copy of the entire object in its cache, it can use the `Range` request-header with a conditional `GET` using either or both of `If-Unmodified-Since` and `If-Match` headers. If the condition fails because an aliased object was updated, the client must make a second request to obtain the entire current object. The `If-Range` header allows a client to "short-circuit" the second request; "if the object is unchanged, send the missing part(s); otherwise, send the entire object."

If the client has no entity tag for an object but has a `Last-Modified` date, it can use that date in an `If-Range` header. Swarm can distinguish between a valid HTTP-date and any form of entity-tag by looking for double quotes. The `If-Range` header should be used together with a `Range` header, and is ignored if the request does not include a `Range` header.

If the entity tag given in the `If-Range` header matches the current primary object UUID or the HTTP-date given is not before the `Last-Modified` date of the object, Swarm provides the specified sub-range of the object using an `HTTP 206 Partial content` response. If the entity tag does not match, Swarm returns the entire object using an `HTTP 200 OK` response.

HTTP 1.0 Caching Headers

In the first version of HTTP, the cache coherency mechanism used time stamps with one-second granularity to decide if a resource was modified and, therefore, required invalidating the cached copy. In addition to the coarse time granularity that can mask changes made in the same second (to aliased objects for example), this approach also requires the client and/or proxy clocks to be reasonably well synchronized with the server clocks.

Warning

Although Swarm supports this coherency method for compatibility reasons, it is not the preferred mechanism because of these issues and is not supported for rapid update use cases. ETag comparisons are recommended for cache coherency on objects that are rapidly updated. The value of each date header adheres to the [Full Date Section 3.3.1](#) of the HTTP/1.1 specification and dates in that format are recognized by Swarm on incoming requests.

Swarm supports the following HTTP/1.0 caching headers:

- Last-Modified
- If-Modified-Since
- If-Unmodified-Since
- Expires

Last-Modified

Swarm returns the Last-Modified header for all **POST**, **PUT**, **COPY**, **APPEND**, **GET**, and **HEAD** operations. The value of the header is exactly the same as the **Castor-System-Created** header for both ordinary objects and aliased objects.

- This is the original object time stamp for ordinary objects.
- This is the server time when the alias was last updated for aliased objects.

Castor-System-Created deprecated

The **Castor-System-Created** header is deprecated, replaced with the more standard **Last-Modified** header. For backward compatibility with previously stored data, Swarm continues to generate both headers and operates as it does now if it encounters an object with a **Castor-System-Created** header, but without a **Last-Modified** header. If a stored object includes both headers, Swarm uses the value of the **Last-Modified** header. A future release ceases generating the deprecated header for newly-stored content.

If-Modified-Since

A Swarm client or proxy can include the **If-Modified-Since** header with a **GET** or **HEAD** method request. All other methods ignore the header when present in the request. The **If-Modified-Since** request header field is used with a **GET** to make it conditional.

Note

If-Modified-Since is for use with GET and HEAD requests (not writes). If specifying a date in the future, Swarm ignores it.

An entity is not returned from the server if the requested object was not modified since the time specified in the **If-Modified-Since** header. Instead, an **HTTP 304 Not Modified** response is returned without any message-body.

See [Section 14.25](#) in the HTTP 1.1 specification for details.



Best practice

If storing frequently updated mutable objects, use ETag comparisons, which offer cache coherency on rapidly updated objects.

If-Unmodified-Since

A Swarm client or proxy can include this header with a **GET**, **PUT**, or **DELETE** method. All other methods ignore this header when present in the request. The **If-Unmodified-Since** request header field is used with a method to make it conditional.

- Swarm performs the requested method as if the **If-Unmodified-Since** header is not present if the requested object is not modified since the time specified in this field.
- Swarm does not perform the requested method, and instead, returns an **HTTP 412 Precondition failed** if the requested object is modified since the specified time.
- The header is ignored if the specified date is invalid.

Expires

Swarm returns an [Expires header](#) if it is persisted with the content. Swarm does not generate an Expires header.

The Expires header field provides the final date and time when the response is considered stale. A stale cache entry may not normally be returned by a cache (either a proxy cache or a user agent cache) unless it is first validated with Swarm (or with an intermediate cache that has a fresh copy of the object). Since Swarm has no information about when an aliased object may be updated and little information about when an object may be deleted, Swarm does not generate an Expires header for any object. Expires are added to the list of persisted headers so applications can supply a hint to caching proxies and clients as to when an object may become stale.

SCS CLI Commands

The Swarm Cluster Services (SCS) configures and manages Swarm storage clusters. This command-line interface helps automate hardware management tasks by enabling script common, high-level management tasks around cluster and node deployment:

Cluster	<ul style="list-style-type: none"> • Create a storage cluster • Read a storage cluster's settings • Update a storage cluster's settings
Chassis	<ul style="list-style-type: none"> • Add a chassis to the cluster • Read a chassis' settings • Update a chassis' settings
Software	<ul style="list-style-type: none"> • Upgrade a storage cluster's software • Add or update a storage license • Update the storage cluster configuration, such as the location of Syslog and NTP servers

SCS command-line interface (CLI) is a set of Python 3 modules; these modules follow a naming convention allowing dynamic discovery, which is then parsed for subcommands. This enables help at any level to learn how to build a command.

CLI Command – The command for the CLI is "`scsctl`", and subcommands are listed in order after. To view settings, a command like "`scsctl settings view`" is used (with arguments as needed).

API Interaction – The CLI uses the SCS API directly as a client. Use the CLI as a set of examples for integrating with the SCS API.

Component Settings – Most setting definitions have a default value. For those depending on *an* environment, the CLI prompts for a value. For secure settings (such as passwords), the CLI does not echo an entry.

- [scsctl](#)
- [scsctl <component>](#)
- [scsctl <component> add](#)
- [scsctl <component> config](#)
- [scsctl <component> config file](#)
- [scsctl <component> config file list](#)
- [scsctl <component> config file set](#)
- [scsctl <component> config file show](#)
- [scsctl <component> config file unset](#)
- [scsctl <component> config list](#)
- [scsctl <component> config set](#)
- [scsctl <component> config show](#)
- [scsctl <component> config unset](#)
- [scsctl <component> group](#)
- [scsctl <component> group add](#)
- [scsctl <component> group list](#)
- [scsctl <component> group makedefault](#)
- [scsctl <component> group remove](#)
- [scsctl <component> group show](#)
- [scsctl <component> group update](#)

- [scsctl <component> instance](#)
- [scsctl <component> instance list](#)
- [scsctl <component> instance remove](#)
- [scsctl <component> instance show](#)
- [scsctl <component> list](#)
- [scsctl <component> show](#)
- [scsctl <component> software](#)
- [scsctl <component> software activate](#)
- [scsctl <component> software list](#)
- [scsctl <component> software show](#)
- [scsctl auth](#)
- [scsctl auth login](#)
- [scsctl auth logout](#)
- [scsctl backup](#)
- [scsctl backup create](#)
- [scsctl backup restore](#)
- [scsctl diagnostics](#)
- [scsctl diagnostics config](#)
- [scsctl diagnostics config scan_missing](#)
- [scsctl init](#)
- [scsctl init configs](#)
- [scsctl init dhcp](#)
- [scsctl init wizard](#)
- [scsctl license](#)
- [scsctl license add](#)
- [scsctl license show](#)
- [scsctl repo](#)
- [scsctl repo component](#)
- [scsctl repo component add](#)
- [scsctl repo component build](#)
- [scsctl repo component delete](#)
- [scsctl repo component list](#)
- [scsctl repo component reload](#)
- [scsctl repo thirdparty](#)
- [scsctl repo thirdparty add](#)
- [scsctl repo third party delete](#)
- [scsctl repo third party list](#)
- [scsctl system](#)
- [scsctl system reset](#)

scsctl

Provides basic control and visibility in to the SCS service.

Usage

```
usage: scsctl [-v | --info-log | --debug-log | --trace-log]
            [--user USER | --token TOKEN]
            options: ...
```

Provides basic control and visibility into the Platform service.

optional arguments:

```
--user USER    User name and (optionally) password. If password is included,
                then this must be in the form "{user_name}:{password}"
                (default: None)
--token TOKEN   Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose   Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                is always sent to stdout. (default: None)
--info-log      Set info-level verbosity (equivalent to -v). (default: None)
--debug-log     Set debug-level verbosity (equivalent to -vv). (default:
                None)
--trace-log     Set trace-level verbosity (equivalent to -vvv). (default:
                None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl auth help'

options:

```
auth           Manages persisted credentials for contacting the Platform
                API.
backup         Manages backing up and restoring the Platform service.
diagnostics    Performs diagnostics on the Platform server setup.
init           Utilities for setting up a Swarm environment.
license        Manages the license for this Swarm ecosystem installation.
repo           Manages the Platform repository. The repository contains
                installed component versions and other third-party software.
system         Performs administrative operations on the system.
<component>   Manage the "<component>" component.
```

scsctl <component>

Manages a component in the Swarm ecosystem.

Usage

```
usage: scsctl <component> [-v | --info-log | --debug-log | --trace-log]
                        [--user USER | --token TOKEN] [--version VERSION]
                        [-b] [--pretty]
options: ...
```

Manage the "<component>" component.

optional arguments:

```
--user USER           User name and (optionally) password. If password is
                        included, then this must be in the form
                        "{user_name}:{password}" (default: None)
--token TOKEN         Authentication token. (default: None)
--version VERSION     A specific installed version to view. (default: None)
-b, --brief           If present, will display component details in compact
                        mode. (default: True)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose         Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                        output is always sent to stdout. (default: None)
--info-log            Set info-level verbosity (equivalent to -v). (default:
                        None)
--debug-log           Set debug-level verbosity (equivalent to -vv). (default:
                        None)
--trace-log           Set trace-level verbosity (equivalent to -vvv). (default:
                        None)
--pretty              If specified, will format output to be more human-
                        friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> add help'

options:

```
add                   Shorthand for component "instance add".
config                Manage configuration for this component.
group                 Manage groups within this component.
instance              Manage instances of this component.
list                  Shorthand for component "instance list".
show                  Display this component's details.
software              Manage software for this component.
```

scsctl <component> add

Shorthand for component "instance add".

Usage

```
usage: scsctl <component> add [-v | --info-log | --debug-log | --trace-log]
                               [--user USER | --token TOKEN] [--show-detail]
                               (-g GROUP_NAME | -d)
                               options: ... name
```

Shorthand for component "instance add".

positional arguments:

name The name of the instance.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

--show-detail If specified, then advanced detail about the state of the added instance will be shown (default behavior is to just show the ID). (default: False)

-g GROUP_NAME, --group GROUP_NAME The name of the group to which the instance will be added. (default: _default)

-d, --default-group If specified, then the instance will be added to the default group. (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> add DUMMY help'

options:

scsctl <component> config

Manage configuration for this component.

Usage

```
usage: scsctl <component> config [-v | --info-log | --debug-log | --trace-log]
                                   [--user USER | --token TOKEN]
                                   [-g GROUP_NAME | -d] [-i INSTANCE] [-a]
                                   [--pretty]
options: ...
```

Manage configuration for this component.

optional arguments:

```
--user USER           User name and (optionally) password. If password is
                       included, then this must be in the form
                       "{user_name}:{password}" (default: None)
--token TOKEN         Authentication token. (default: None)
-g GROUP_NAME, --group GROUP_NAME
                       The name of the group for which to list. (default:
                       None)
-d, --default-group  If specified, then the default group will be used for
                       listing. (default: None)
-i INSTANCE, --instance INSTANCE
                       The name or ID of the instance for which to list.
                       (default: None)
-a, --include-advanced
                       If provided, then advanced settings will be included
                       in the listing. (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose         Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                       output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                       None)
--debug-log         Set debug-level verbosity (equivalent to -vv).
                       (default: None)
--trace-log         Set trace-level verbosity (equivalent to -vvv).
                       (default: None)
--pretty            If specified, will format output to be more human-
                       friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> config file help'

options:

```
file                Manages component configuration files.
list                Lists configuration settings for the current
                       component.
set                 Updates a configuration setting for the current
                       component.
show               Displays a configuration setting for the current
                       component.
unset              Removes a previously-set configuration setting value
                       of the given component.
```

scsctl <component> config file

Manages component configuration files.

Usage

```
usage: scsctl <component> config file
        [-v | --info-log | --debug-log | --trace-log]
        [--user USER | --token TOKEN]
        [-g GROUP_NAME | -d] [-i INSTANCE]
options: ...
```

Manages component configuration files.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-g GROUP_NAME, --group GROUP_NAME
                    The name of the group for which to list. (default:
                    None)
-d, --default-group If specified, then the default group will be used for
                    listing. (default: None)
-i INSTANCE, --instance INSTANCE
                    The name or ID of the instance for which to list.
                    (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> config file list help'

options:

```
list                Lists configuration files for the current component.
set                 Updates a configuration file for the current
                    component.
show                Shows a configuration file for the current component.
unset              Removes a customized configuration file for the
                    current component.
```

scsctl <component> config file list

Lists configuration files for the current component.

Usage

```
usage: scsctl <component> config file list
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     [-g GROUP_NAME | -d] [-i INSTANCE]
                                     options: ...
```

Lists configuration files for the current component.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-g GROUP_NAME, --group GROUP_NAME
                    The name of the group for which to list. (default:
                    None)
-d, --default-group If specified, then the default group will be used for
                    listing. (default: None)
-i INSTANCE, --instance INSTANCE
                    The name or ID of the instance for which to list.
                    (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> config file list DUMMY help'

options:

scsctl <component> config file set

Updates a configuration file for the current component.

Usage

```
usage: scsctl <component> config file set
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     (-g GROUP_NAME | -d) [-i INSTANCE]
                                     (-f FILE | -u URL)
                                     options: ... name
```

Updates a configuration file for the current component.

positional arguments:

name The name of the configuration file to update.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

-g GROUP_NAME, --group GROUP_NAME The name of the group where the update should occur. (default: None)

-d, --default-group If specified, then the default group will be used for the update. (default: None)

-i INSTANCE, --instance INSTANCE The name or ID of the instance where the update should occur. (default: None)

-f FILE, --file FILE Path to config file to use for the update, or '-' to read from stdin. (default: None)

-u URL, --url URL URL of config file to use for the update. (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> config file set DUMMY help'

options:

scsctl <component> config file show

Shows a configuration file for the current component.

Usage

```
usage: scsctl <component> config file show
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     [-g GROUP_NAME | -d] [-i INSTANCE]
                                     [-r]
                                     options: ... name
```

Shows a configuration file for the current component.

positional arguments:

name The name of the configuration file to show.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

-g GROUP_NAME, --group GROUP_NAME The name of the group for which to show. (default: None)

-d, --default-group If specified, then the default group will be used for showing. (default: None)

-i INSTANCE, --instance INSTANCE The name or ID of the instance for which to show. (default: None)

-r, --raw Whether or not to show the raw (un-rendered) configuration file. Only applies if "instance" is specified. (default: False)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> config file show DUMMY help'

options:

scsctl <component> config file unset

Removes a customized configuration file for the current component.

Usage

```
usage: scsctl <component> config file unset
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     (-g GROUP_NAME | -d) [-i INSTANCE]
                                     options: ... name
```

Removes a customized configuration file for the current component.

positional arguments:

name The name of the customized configuration file to remove.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

-g GROUP_NAME, --group GROUP_NAME The name of the group where the removal should occur. (default: None)

-d, --default-group If specified, then the default group will be used for removing. (default: None)

-i INSTANCE, --instance INSTANCE The name or ID of the instance where the removal should occur. (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> config file unset DUMMY help'

options:

scsctl <component> config list

Lists configuration settings for the current component.

Usage

```
usage: scsctl <component> config list
        [-v | --info-log | --debug-log | --trace-log]
        [--user USER | --token TOKEN]
        [-g GROUP_NAME | -d] [-i INSTANCE] [-a]
        [--pretty]
options: ...
```

Lists configuration settings for the current component.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-g GROUP_NAME, --group GROUP_NAME
                    The name of the group for which to list. (default:
                    None)
-d, --default-group If specified, then the default group will be used for
                    listing. (default: None)
-i INSTANCE, --instance INSTANCE
                    The name or ID of the instance for which to list.
                    (default: None)
-a, --include-advanced
                    If provided, then advanced settings will be included
                    in the listing. (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
--pretty             If specified, will format output to be more human-
                    friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> config list DUMMY help'

options:

scsctl <component> config set

Updates a configuration setting for the current component.

Usage

```
usage: scsctl <component> config set
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     (-g GROUP_NAME | -d) [-i INSTANCE]
                                     [-f FILE] [-u URL]
                                     options: ... [setting_assignment]
```

Updates a configuration setting for the current component.

positional arguments:

```
setting_assignment    A setting assignment in the form of:
                     setting_name=new_value (default: None)
```

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                     included, then this must be in the form
                     "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-g GROUP_NAME, --group GROUP_NAME
                     The name of the group where the update should occur.
                     (default: None)
-d, --default-group If specified, then the default group will be used for
                     updating. (default: None)
-i INSTANCE, --instance INSTANCE
                     The name or ID of the instance where the update should
                     occur. (default: None)
-f FILE, --file FILE Path to JSON file with object of name/value pairs to
                     use for the update, or '-' to read from stdin.
                     (default: None)
-u URL, --url URL    URL of JSON file with object of name/value pairs to
                     use for the update. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                     output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                     None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                     (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                     (default: None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> config set DUMMY help'

options:

scsctl <component> config show

Displays a configuration setting for the current component.

Usage

```
usage: scsctl <component> config show
      [-v | --info-log | --debug-log | --trace-log]
      [--user USER | --token TOKEN]
      [-g GROUP_NAME | -d] [-i INSTANCE]
      [--show-detail] [--pretty]
options: ... name
```

Displays a configuration setting for the current component.

positional arguments:

name The name of the configuration setting to show.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

-g GROUP_NAME, --group GROUP_NAME The name of the group for which to show. (default: None)

-d, --default-group If specified, then the default group will be used for showing. (default: None)

-i INSTANCE, --instance INSTANCE The name or ID of the instance for which to show. (default: None)

--show-detail If specified, then advanced detail about the state of the configuration setting will be shown (default behavior is to just show the value). (default: False)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

--pretty If specified, will format output to be more human-friendly. (default: False)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> config show DUMMY help'

options:

scsctl <component> config unset

Removes a previously-set configuration setting value of the given component.

Usage

```
usage: scsctl <component> config unset
                                [-v | --info-log | --debug-log | --trace-log]
                                [--user USER | --token TOKEN]
                                (-g GROUP_NAME | -d) [-i INSTANCE]
                                options: ... name
```

Removes a previously-set configuration setting value of the given component.

positional arguments:

name The name of the configuration setting to un-set.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

-g GROUP_NAME, --group GROUP_NAME The name of the group for which to un-set. (default: None)

-d, --default-group If specified, then the default group will be used for removing. (default: None)

-i INSTANCE, --instance INSTANCE The name or ID of the instance for which to un-set. (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> config unset DUMMY help'

options:

scsctl <component> group

Manage groups within this component.

Usage

```
usage: scsctl <component> group [-v | --info-log | --debug-log | --trace-log]
                                   [--user USER | --token TOKEN]
                                   [--no-default-marker] [--pretty]
options: ...
```

Manage groups within this component.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                      included, then this must be in the form
                      "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
--no-default-marker  If specified, hides the marker designating which group
                      is the default. (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                      output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                      None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                      (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                      (default: None)
--pretty             If specified, will format output to be more human-
                      friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> group add help'

options:

```
add                 Adds a group to the given component.
list                Lists groups of the given component.
makedefault         Marks a group as the default group in the given
                      component.
remove              Removes a group from the given component.
show                Displays information about a group of the given
                      component.
update              Updates a group in the given component.
```

scsctl <component> group add

Adds a group to the given component.

Usage

```
usage: scsctl <component> group add
                                [-v | --info-log | --debug-log | --trace-log]
                                [--user USER | --token TOKEN]
                                [-d DESCRIPTION] [--make-default]
                                options: ... name
```

Adds a group to the given component.

positional arguments:

name The name of the group to be added.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

-d DESCRIPTION, --description DESCRIPTION A description of the cluster (purpose, location, etc.) (default:)

--make-default If given, then the newly added group will be marked as the default. Note that if this is the first group in the component it will automatically be marked as default. (default: False)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> group add DUMMY help'

options:

scsctl <component> group list

Lists groups of the given component.

Usage

```
usage: scsctl <component> group list
      [-v | --info-log | --debug-log | --trace-log]
      [--user USER | --token TOKEN]
      [--no-default-marker] [--pretty]
      options: ...
```

Lists groups of the given component.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
--no-default-marker  If specified, hides the marker designating which group
                    is the default. (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
--pretty             If specified, will format output to be more human-
                    friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> group list DUMMY help'

options:

scsctl <component> group makedefault

Marks a group as the default group in the given component.

Usage

```
usage: scsctl <component> group makedefault
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     options: ... name
```

Marks a group as the default group in the given component.

positional arguments:

name The name of the group to be marked as default.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)
--token TOKEN Authentication token. (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)
--info-log Set info-level verbosity (equivalent to -v). (default: None)
--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)
--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> group makedefault DUMMY help'

options:

scsctl <component> group remove

Removes a group from the given component.

Usage

```
usage: scsctl <component> group remove
                                [-v | --info-log | --debug-log | --trace-log]
                                [--user USER | --token TOKEN] [--force]
                                [-y]
                                options: ... name
```

Removes a group from the given component.

positional arguments:

name The name of the group to remove.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)
 --token TOKEN Authentication token. (default: None)
 --force If set, forces the group to be removed even if it is marked as the default group. (default: False)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)
 --info-log Set info-level verbosity (equivalent to -v). (default: None)
 --debug-log Set debug-level verbosity (equivalent to -vv). (default: None)
 --trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)
 -y, --yes If specified, will bypass all confirmation prompts. (default: False)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> group remove DUMMY help'

options:

scsctl <component> group show

Displays information about a group of the given component.

Usage

```
usage: scsctl <component> group show
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN] [--pretty]
options: ... name
```

Displays information about a group of the given component.

positional arguments:

name The name of the group to show.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)
 --token TOKEN Authentication token. (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)
 --info-log Set info-level verbosity (equivalent to -v). (default: None)
 --debug-log Set debug-level verbosity (equivalent to -vv). (default: None)
 --trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)
 --pretty If specified, will format output to be more human-friendly. (default: False)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> group show DUMMY help'

options:

scsctl <component> group update

Updates a group in the given component.

Usage

```
usage: scsctl <component> group update
                                [-v | --info-log | --debug-log | --trace-log]
                                [--user USER | --token TOKEN]
                                [-d DESCRIPTION]
                                options: ... name
```

Updates a group in the given component.

positional arguments:

name The name of the group to be updated.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

-d DESCRIPTION, --description DESCRIPTION A description of the cluster (purpose, location, etc.) (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> group update DUMMY help'

options:

scsctl <component> instance

Manage instances of this component.

Usage

```
usage: scsctl <component> instance
      [-v | --info-log | --debug-log | --trace-log]
      [--user USER | --token TOKEN]
      [-g GROUP_NAME | -d]
      [-s {group,id,ip_address,name}] [--pretty]
options: ...
```

Manage instances of this component.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-g GROUP_NAME, --group GROUP_NAME
                    The name of the group to list. If missing, then all
                    instances in all groups will be listed. (default:
                    None)
-d, --default-group If specified, then the default group will be used for
                    listing. (default: None)
-s {group,id,ip_address,name}, --sort-by {group,id,ip_address,name}
                    The column to use for sorting. (default: name)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log          Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log         Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log         Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
--pretty            If specified, will format output to be more human-
                    friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> instance add help'

options:

```
list                Lists instances of the given component.
remove              Removes an instance of the given component.
show                Displays information about an instance of the given
                    component.
```

scsctl <component> instance list

Lists instances of the given component.

Usage

```
usage: scsctl <component> instance list
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     [-g GROUP_NAME | -d]
                                     [-s {group,id,ip_address,name}]
                                     [--pretty]
                                     options: ...
```

Lists instances of the given component.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-g GROUP_NAME, --group GROUP_NAME
                    The name of the group to list. If missing, then all
                    instances in all groups will be listed. (default:
                    None)
-d, --default-group If specified, then the default group will be used for
                    listing. (default: None)
-s {group,id,ip_address,name}, --sort-by {group,id,ip_address,name}
                    The column to use for sorting. (default: name)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
--pretty             If specified, will format output to be more human-
                    friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> instance list DUMMY help'

options:

scsctl <component> instance remove

Removes an instance of the given component.

Usage

```
usage: scsctl <component> instance remove
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     (-g GROUP_NAME | -d)
                                     options: ... instance
```

Removes an instance of the given component.

positional arguments:

instance The name or ID of the instance to remove.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

-g GROUP_NAME, --group GROUP_NAME The name of the group from which to remove. (default: None)

-d, --default-group If specified, then the instance will be looked up within the default group. (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> instance remove DUMMY help'

options:

scsctl <component> instance show

Displays information about an instance of the given component.

Usage

```
usage: scsctl <component> instance show
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     (-g GROUP_NAME | -d) [--pretty]
                                     options: ... instance
```

Displays information about an instance of the given component.

positional arguments:

instance The name or ID of the instance to show.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

-g GROUP_NAME, --group GROUP_NAME The name of the group containing the instance. (default: None)

-d, --default-group If specified, then the instance will be looked up within the default group. (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

--pretty If specified, will format output to be more human-friendly. (default: False)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> instance show DUMMY help'

options:

scsctl <component> list

Shorthand for component "instance list".

Usage

```
usage: scsctl <component> list [-v | --info-log | --debug-log | --trace-log]
                                [--user USER | --token TOKEN]
                                [-g GROUP_NAME | -d]
                                [-s {group,id,ip_address,name}] [--pretty]
                                options: ...
```

Shorthand for component "instance list".

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-g GROUP_NAME, --group GROUP_NAME
                    The name of the group to list. If missing, then all
                    instances in all groups will be listed. (default:
                    None)
-d, --default-group If specified, then the default group will be used for
                    listing. (default: None)
-s {group,id,ip_address,name}, --sort-by {group,id,ip_address,name}
                    The column to use for sorting. (default: group)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
--pretty             If specified, will format output to be more human-
                    friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> list DUMMY help'

options:

scsctl <component> show

Display this component's details.

Usage

```
usage: scsctl <component> show [-v | --info-log | --debug-log | --trace-log]
                                [--user USER | --token TOKEN]
                                [--version VERSION] [-b] [--pretty]
options: ...
```

Display this component's details.

optional arguments:

```
--user USER      User name and (optionally) password. If password is
                  included, then this must be in the form
                  "{user_name}:{password}" (default: None)
--token TOKEN    Authentication token. (default: None)
--version VERSION A specific installed version to view. (default: None)
-b, --brief      If present, will display component details in compact
                  mode. (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose      Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                  output is always sent to stdout. (default: None)
--info-log        Set info-level verbosity (equivalent to -v). (default:
                  None)
--debug-log       Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log       Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
--pretty          If specified, will format output to be more human-
                  friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> show DUMMY help'

options:

scsctl <component> software

Manage software for this component.

Usage

```
usage: scsctl <component> software
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN] [--pretty]
                                     options: ...
```

Manage software for this component.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN     Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose     Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log        Set info-level verbosity (equivalent to -v). (default: None)
--debug-log       Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log       Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
--pretty          If specified, will format output to be more human-friendly.
                  (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> software activate help'

options:

```
activate         Activates an installed version of a component in the Swarm
                  ecosystem.
list             Lists installed software versions for a component in the
                  Swarm ecosystem.
show            Displays software information for an installed version of the
                  component.
```

scsctl <component> software activate

Activates an installed version of a component in the Swarm ecosystem.

Usage

```
usage: scsctl <component> software activate
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     options: ... [version]
```

Activates an installed version of a component in the Swarm ecosystem.

positional arguments:

```
version    The installed version to activate. Consult the list of
            installed versions for valid values. (default: None)
```

optional arguments:

```
--user USER  User name and (optionally) password. If password is included,
              then this must be in the form "{user_name}:{password}"
              (default: None)
--token TOKEN Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose  Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
               is always sent to stdout. (default: None)
--info-log    Set info-level verbosity (equivalent to -v). (default: None)
--debug-log   Set debug-level verbosity (equivalent to -vv). (default:
               None)
--trace-log   Set trace-level verbosity (equivalent to -vvv). (default:
               None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> software activate DUMMY help'

options:

scsctl <component> software list

Lists installed software versions for a component in the Swarm ecosystem.

Usage

```
usage: scsctl <component> software list
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     [--pretty]
                                     options: ...
```

Lists installed software versions for a component in the Swarm ecosystem.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN    Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose    Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log       Set info-level verbosity (equivalent to -v). (default: None)
--debug-log      Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log      Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
--pretty         If specified, will format output to be more human-friendly.
                  (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> software list DUMMY help'

options:

scsctl <component> software show

Displays software information for an installed version of the component.

Usage

```
usage: scsctl <component> software show
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     [--pretty]
                                     options: ... version
```

Displays software information for an installed version of the component.

positional arguments:

version The installed version to view.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)
 --token TOKEN Authentication token. (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)
 --info-log Set info-level verbosity (equivalent to -v). (default: None)
 --debug-log Set debug-level verbosity (equivalent to -vv). (default: None)
 --trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)
 --pretty If specified, will format output to be more human-friendly. (default: False)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl <component> software show DUMMY help'

options:

scsctl auth

Manages persisted credentials for contacting the SCS API.

Usage

```
usage: scsctl auth [-v | --info-log | --debug-log | --trace-log]
                  [--user USER | --token TOKEN]
                  options: ...
```

Manages persisted credentials for contacting the Platform API.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN     Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose     Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log        Set info-level verbosity (equivalent to -v). (default: None)
--debug-log       Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log       Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl auth login help'

options:

```
login            Stores new credentials for contacting the Platform API.
logout           Clears previously-stored credentials for contacting the
                  Platform API.
```

scsctl auth login

Stores new credentials for contacting the SCS API.

Usage

```
usage: scsctl auth login [-v | --info-log | --debug-log | --trace-log]
                        [--user USER | --token TOKEN]
                        options: ...
```

Stores new credentials for contacting the Platform API.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN     Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose     Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log        Set info-level verbosity (equivalent to -v). (default: None)
--debug-log       Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log       Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl auth login DUMMY help'

options:

scsctl auth logout

Clears previously-stored credentials for contacting the SCS API.

Usage

```
usage: scsctl auth logout [-v | --info-log | --debug-log | --trace-log]
                        [--user USER | --token TOKEN]
                        options: ...
```

Clears previously-stored credentials for contacting the Platform API.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN     Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose     Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log        Set info-level verbosity (equivalent to -v). (default: None)
--debug-log       Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log       Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl auth logout DUMMY help'

options:

scsctl backup

Manages backing up and restoring the SCS service.

Usage

```
usage: scsctl backup [-v | --info-log | --debug-log | --trace-log]
                  [--user USER | --token TOKEN] [--no-repo] [-o OUTPUT]
                  options: ...
```

Manages backing up and restoring the Platform service.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
--no-repo            If specified, then repo data (binaries, etc.) will be
                    excluded from the backup. (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
-o OUTPUT, --output OUTPUT
                    Where to send the result of the command (non-verbose
                    output; default is stdout, same as specifying "-").
                    (default: -)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl backup create help'

options:

```
create              Performs a backup of the Platform service.
restore             Performs a restore of the Platform service from a
                    backup file.
```

scsctl backup create

Performs a backup of the SCS service.

Usage

```
usage: scsctl backup create [-v | --info-log | --debug-log | --trace-log]
                          [--user USER | --token TOKEN] [--no-repo]
                          [-o OUTPUT]
                          options: ...
```

Performs a backup of the Platform service.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
--no-repo            If specified, then repo data (binaries, etc.) will be
                    excluded from the backup. (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
-o OUTPUT, --output OUTPUT
                    Where to send the result of the command (non-verbose
                    output; default is stdout, same as specifying "-").
                    (default: -)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl backup create DUMMY help'

options:

scsctl backup restore

Performs a restore of the SCS service from a backup file.

Usage

```
usage: scsctl backup restore [-v | --info-log | --debug-log | --trace-log]
                             [--user USER | --token TOKEN] [-o OUTPUT]
                             options: ... backup_file
```

Performs a restore of the Platform service from a backup file.

positional arguments:

backup_file Path to backup image to be restored.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

-o OUTPUT, --output OUTPUT Where to send the result of the command (non-verbose output; default is stdout, same as specifying "-"). (default: -)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl backup restore DUMMY help'

options:

scsctl diagnostics

Performs diagnostics on the SCS server setup.

Usage

```
usage: scsctl diagnostics [-v | --info-log | --debug-log | --trace-log]
                        [--user USER | --token TOKEN] [--quick]
                        [--scan-only] [--pretty]
options: ...
```

Performs diagnostics on the Platform server setup.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN    Authentication token. (default: None)
--quick          Only performs a quick scan of potential configuration issues
                  (implies --scan-only). (default: False)
--scan-only      Only performs a quick scan of potential configuration issues.
                  (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose    Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log       Set info-level verbosity (equivalent to -v). (default: None)
--debug-log      Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log      Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
--pretty         If specified, will format output to be more human-friendly.
                  (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl diagnostics config help'

options:

```
config          Checks configuration across the ecosystem.
```

scsctl diagnostics config

Checks configuration across the ecosystem.

Usage

```
usage: scsctl diagnostics config [-v | --info-log | --debug-log | --trace-log]
                                [--user USER | --token TOKEN] [--quick]
                                [--scan-only] [--pretty]
options: ...
```

Checks configuration across the ecosystem.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN     Authentication token. (default: None)
--quick           Only performs a quick scan of potential configuration issues
                  (implies --scan-only). (default: False)
--scan-only      Only performs a quick scan of potential configuration issues.
                  (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose    Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log       Set info-level verbosity (equivalent to -v). (default: None)
--debug-log      Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log      Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
--pretty         If specified, will format output to be more human-friendly.
                  (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl diagnostics config scan_missing help'

options:

```
scan_missing     Checks for missing configuration values.
```

scsctl diagnostics config scan_missing

Checks for missing configuration values.

Usage

```
usage: scsctl diagnostics config scan_missing
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN]
                                     [--quick] [--scan-only]
                                     [--pretty]
                                     options: ...
```

Checks for missing configuration values.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN    Authentication token. (default: None)
--quick          Only performs a quick scan of potential configuration issues
                  (implies --scan-only). (default: False)
--scan-only      Only performs a quick scan of potential configuration issues.
                  (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose    Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log       Set info-level verbosity (equivalent to -v). (default: None)
--debug-log      Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log      Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
--pretty         If specified, will format output to be more human-friendly.
                  (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl diagnostics config scan_missing DUMMY help'

options:

scsctl init

Utilities for setting up a Swarm environment.

Usage

```
usage: scsctl init [-v | --info-log | --debug-log | --trace-log]
                  [--user USER | --token TOKEN] [--force] [--next-steps]
                  [--allow-all-system-updates]
                  [--details-path [DETAILS_PATH]]
options: ...
```

Utilities for setting up a Swarm environment.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
--force              If set, forces wizard to restart the entire process.
                    (default: False)
--next-steps         If specified, will only display the next steps to run
                    after this wizard and exit. (default: False)
--allow-all-system-updates
                    WARNING: Setting this will BYPASS ALL PROMPTS warning
                    of changes to the system! Use with caution. (default:
                    False)
--details-path [DETAILS_PATH]
                    If given, then all details of the init process will be
                    captured in the specified file. To disable, specify
                    this argument but do not provide a value. (default:
                    ./init-details.log)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl init configs help'

options:

```
configs             Performs an import of config values for multiple
                    components into the default group for each. Each
                    component's config values are in a file for that
                    component, with the component name as the base
                    ("your_component.cfg", "your_coomponent.json", etc.)
dhcp                Configures on-box DHCP service for use with Platform.
wizard              Walks the user through the init process to get the
                    environment set up.
```

scsctl init configs

Performs an import of config values for multiple components into the default group for each. Each component's config values are in a file for that component, with the component name as the base ("your_component.cfg", "your_component.json", etc.)

Usage

```
usage: scsctl init configs [-v | --info-log | --debug-log | --trace-log]
      [--user USER | --token TOKEN] [-d DIRECTORY]
      options: ...
```

Performs an import of config values for multiple components into the default group for each. Each component's config values are in a file for that component, with the component name as the base ("your_component.cfg", "your_coomponent.json", etc.)

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-d DIRECTORY, --directory DIRECTORY
                    Path to the directory where the files are stored.
                    (default: .)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log         Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log         Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl init configs DUMMY help'

options:

scsctl init dhcp

Configures on-box DHCP service for use with SCS.

Usage

```
usage: scsctl init dhcp [-v | --info-log | --debug-log | --trace-log]
                        [--user USER | --token TOKEN] --dns-domain DNS_DOMAIN
                        --dns-servers DNS_SERVERS --ntp-servers NTP_SERVERS
                        [--dhcp-lease-default DHCP_LEASE_DEFAULT]
                        [--dhcp-lease-max DHCP_LEASE_MAX]
                        [--dhcp-reserve-lower DHCP_RESERVE_LOWER]
                        [--dhcp-reserve-upper DHCP_RESERVE_UPPER]
                        [--dhcp-transient-percent DHCP_TRANSIENT_PERCENT]
                        options: ...
```

Configures on-box DHCP service for use with Platform.

optional arguments:

```
--user USER           User name and (optionally) password. If password is
                        included, then this must be in the form
                        "{user_name}:{password}" (default: None)
--token TOKEN         Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose         Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                        output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                        None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                        (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                        (default: None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl init dhcp DUMMY help'

options:

Related Services:

```
--dns-domain DNS_DOMAIN
                        DNS domain name (default: None)
--dns-servers DNS_SERVERS
                        DNS servers to be used, comma-delimited (default:
                        None)
--ntp-servers NTP_SERVERS
                        NTP servers to be used, comma-delimited (default:
                        None)
```

DHCP-specific Settings:

Specifies timeouts and pool ranges within the subnet. Note that at least one of dhcp-reserve-lower or dhcp-reserve-upper MUST be set.

```
--dhcp-lease-default DHCP_LEASE_DEFAULT
                        Default DHCP lease time for storage nodes (default:
                        172800)
--dhcp-lease-max DHCP_LEASE_MAX
                        Maximum allowed DHCP lease time for storage nodes
                        (default: 604800)
--dhcp-reserve-lower DHCP_RESERVE_LOWER
                        Number of IP addresses to reserve in lower subnet
                        range (default: 0)
--dhcp-reserve-upper DHCP_RESERVE_UPPER
                        Number of IP addresses to reserve in upper subnet
                        range (default: 0)
--dhcp-transient-percent DHCP_TRANSIENT_PERCENT
                        Percentage of DHCP managed range for transient clients
                        (default: 50)
```

scsctl init wizard

Walks the user through the init process to get the environment set up.

Usage

```
usage: scsctl init wizard [-v | --info-log | --debug-log | --trace-log]
                        [--user USER | --token TOKEN] [--force]
                        [--next-steps] [--allow-all-system-updates]
                        [--details-path [DETAILS_PATH]]
options: ...
```

Walks the user through the init process to get the environment set up.

optional arguments:

```
--user USER           User name and (optionally) password. If password is
                        included, then this must be in the form
                        "{user_name}:{password}" (default: None)
--token TOKEN         Authentication token. (default: None)
--force               If set, forces wizard to restart the entire process.
                        (default: False)
--next-steps          If specified, will only display the next steps to run
                        after this wizard and exit. (default: False)
--allow-all-system-updates
                        WARNING: Setting this will BYPASS ALL PROMPTS warning
                        of changes to the system! Use with caution. (default:
                        False)
--details-path [DETAILS_PATH]
                        If given, then all details of the init process will be
                        captured in the specified file. To disable, specify
                        this argument but do not provide a value. (default:
                        ./init-details.log)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose         Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                        output is always sent to stdout. (default: None)
--info-log            Set info-level verbosity (equivalent to -v). (default:
                        None)
--debug-log           Set debug-level verbosity (equivalent to -vv).
                        (default: None)
--trace-log           Set trace-level verbosity (equivalent to -vvv).
                        (default: None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl init wizard DUMMY help'

options:

scsctl license

Manages the license for this Swarm ecosystem installation.

Usage

```
usage: scsctl license [-v | --info-log | --debug-log | --trace-log]
                    [--user USER | --token TOKEN] [--pretty]
                    options: ...
```

Manages the license for this Swarm ecosystem installation.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN     Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose    Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log       Set info-level verbosity (equivalent to -v). (default: None)
--debug-log      Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log      Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
--pretty         If specified, will format output to be more human-friendly.
                  (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl license add help'

options:

```
add             Installs a new Swarm license into the Platform service.
show            Displays the license installed in the Platform service.
```

scsctl license add

Installs a new Swarm license into the SCS service.

Usage

```
usage: scsctl license add [-v | --info-log | --debug-log | --trace-log]
                        [--user USER | --token TOKEN] (-f FILE | -u URL)
                        options: ...
```

Installs a new Swarm license into the Platform service.

optional arguments:

```
--user USER           User name and (optionally) password. If password is
                        included, then this must be in the form
                        "{user_name}:{password}" (default: None)
--token TOKEN         Authentication token. (default: None)
-f FILE, --file FILE Path to license file to install, or '-' to read from
                        stdin. (default: None)
-u URL, --url URL     URL of license file to install. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose         Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                        output is always sent to stdout. (default: None)
--info-log            Set info-level verbosity (equivalent to -v). (default:
                        None)
--debug-log           Set debug-level verbosity (equivalent to -vv).
                        (default: None)
--trace-log           Set trace-level verbosity (equivalent to -vvv).
                        (default: None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl license add DUMMY help'

options:

scsctl license show

Displays the license installed in the SCS service.

Usage

```
usage: scsctl license show [-v | --info-log | --debug-log | --trace-log]
                          [--user USER | --token TOKEN] [--pretty]
                          options: ...
```

Displays the license installed in the Platform service.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN     Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose     Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log        Set info-level verbosity (equivalent to -v). (default: None)
--debug-log       Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log       Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
--pretty          If specified, will format output to be more human-friendly.
                  (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl license show DUMMY help'

options:

scsctl repo

Manages the SCS repository. The repository contains installed component versions and other third-party software.

Usage

```
usage: scsctl repo [-v | --info-log | --debug-log | --trace-log]
                [--user USER | --token TOKEN] [-t] [-c]
                [-s {name,bytes,md5}] [--pretty]
options: ...
```

Manages the Platform repository. The repository contains installed component versions and other third-party software.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-t, --table          If present, will list all installed component versions
                    in a tabular view (default is a compact view).
                    (default: False)
-c, --csv            If present, will list in CSV format. (default: False)
-s {name,bytes,md5}, --sort-by {name,bytes,md5}
                    The column to use for sorting. (default: name)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log         Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log         Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
--pretty            If specified, will format output to be more human-
                    friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl repo component help'

options:

```
component          Manages component installations within the Platform
                    repository.
thirdparty         Manages third-party software installations within the
                    Platform repository.
```

scsctl repo component

Manages component installations within the SCS repository.

Usage

```
usage: scsctl repo component [-v | --info-log | --debug-log | --trace-log]
                             [--user USER | --token TOKEN] [-t | -c]
                             [-s {name,is_active}] [--pretty]
                             options: ...
```

Manages component installations within the Platform repository.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-t, --table          If present, will list all installed component versions
                    in a tabular view (default is a compact view).
                    (default: False)
-c, --csv            If present, will list all installed component versions
                    in CSV format. (default: False)
-s {name,is_active}, --sort-by {name,is_active}
                    The column to use for sorting (table and csv output
                    only). (default: name)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
--pretty             If specified, will format output to be more human-
                    friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl repo component add help'

options:

```
add                 Adds a component version to the Platform repository.
build               Builds a component bundle that can be imported into
                    the Platform repository.
delete              Removes a component version from the Platform
                    repository.
list                Lists installed component versions within the Platform
                    repository. The default view is a compact listing of
                    active and inactive components.
reload              Reloads the on-disk component registry.
```

scsctl repo component add

Adds a component version to the SCS repository.

Usage

```
usage: scsctl repo component add [-v | --info-log | --debug-log | --trace-log]
                                [--user USER | --token TOKEN]
                                (-f FILE | -u URL) [--force]
                                [-s | --prompt-for-defaults]
                                options: ...
```

Adds a component version to the Platform repository.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-f FILE, --file FILE Path to component bundle to be added to the
                    repository, or '-' to read from stdin. (default: None)
-u URL, --url URL    URL of component bundle to be added to the repository.
                    (default: None)
--force              If set, forces overwrite of a previously-installed
                    matching component version. (default: False)
-s, --silent         If set, disables prompting for missing configuration
                    value defaults. (default: False)
--prompt-for-defaults
                    If set, will prompt for missing configuration
                    defaults, even if those defaults had been previously
                    set during a prior installation (normally previously-
                    set defaults are automatically used). (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl repo component add DUMMY help'

options:

scsctl repo component build

Builds a component bundle that can be imported into the SCS repository.

Usage

```
usage: scsctl repo component build
      [-v | --info-log | --debug-log | --trace-log]
      [--user USER | --token TOKEN] -d
      DEFINITION_PATH -c SETTINGS_PATH
      [-f TEMPLATE_PATHS] [-s BINARY_PATHS]
      [-o OUTPUT]
      options: ...
```

Builds a component bundle that can be imported into the Platform repository.

required arguments:

```
-d DEFINITION_PATH, --with-definition DEFINITION_PATH
    Path to component definition YAML. (default: None)
-c SETTINGS_PATH, --with-config-definitions SETTINGS_PATH
    Path to configuration definitions. (default: None)
```

optional arguments:

```
--user USER
    User name and (optionally) password. If password is
    included, then this must be in the form
    "{user_name}:{password}" (default: None)
--token TOKEN
    Authentication token. (default: None)
-f TEMPLATE_PATHS, --with-config-file TEMPLATE_PATHS
    Path to a configuration file to add to the bundle.
    (default: None)
-s BINARY_PATHS, --with-software-file BINARY_PATHS
    Path to a software file to add to the bundle.
    (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose
    Increase output verbosity (ex: -v, -vv, -vvv). Verbose
    output is always sent to stdout. (default: None)
--info-log
    Set info-level verbosity (equivalent to -v). (default:
    None)
--debug-log
    Set debug-level verbosity (equivalent to -vv).
    (default: None)
--trace-log
    Set trace-level verbosity (equivalent to -vvv).
    (default: None)
-o OUTPUT, --output OUTPUT
    Where to send the result of the command (non-verbose
    output; default is stdout, same as specifying "-").
    (default: -)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl repo component build DUMMY help'

options:

scsctl repo component delete

Removes a component version from the SCS repository.

Usage

```
usage: scsctl repo component delete
                                [-v | --info-log | --debug-log | --trace-log]
                                [--user USER | --token TOKEN] [--force]
                                [-y]
                                options: ... name version
```

Removes a component version from the Platform repository.

positional arguments:

```
name          The name of the component to remove.
version       The version of the component to remove.
```

optional arguments:

```
--user USER  User name and (optionally) password. If password is included,
              then this must be in the form "{user_name}:{password}"
              (default: None)
--token TOKEN Authentication token. (default: None)
--force       If set, forces removal of requested version, even if it is
              the active version. (default: False)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose  Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
              is always sent to stdout. (default: None)
--info-log     Set info-level verbosity (equivalent to -v). (default: None)
--debug-log    Set debug-level verbosity (equivalent to -vv). (default:
              None)
--trace-log    Set trace-level verbosity (equivalent to -vvv). (default:
              None)
-y, --yes      If specified, will bypass all confirmation prompts. (default:
              False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl repo component delete DUMMY help'

options:

scsctl repo component list

Lists installed component versions within the SCS repository. The default view is a compact listing of active and inactive components.

Usage

```
usage: scsctl repo component list
      [-v | --info-log | --debug-log | --trace-log]
      [--user USER | --token TOKEN] [-t | -c]
      [-s {name,is_active}] [--pretty]
      options: ...
```

Lists installed component versions within the Platform repository. The default view is a compact listing of active and inactive components.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-t, --table          If present, will list all installed component versions
                    in a tabular view (default is a compact view).
                    (default: False)
-c, --csv            If present, will list all installed component versions
                    in CSV format. (default: False)
-s {name,is_active}, --sort-by {name,is_active}
                    The column to use for sorting (table and csv output
                    only). (default: name)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
--pretty             If specified, will format output to be more human-
                    friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl repo component list DUMMY help'

options:

scsctl repo component reload

Reloads the on-disk component registry.

Usage

```
usage: scsctl repo component reload
                                     [-v | --info-log | --debug-log | --trace-log]
                                     [--user USER | --token TOKEN] [-y]
options: ...
```

Reloads the on-disk component registry.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN     Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose     Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log        Set info-level verbosity (equivalent to -v). (default: None)
--debug-log       Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log       Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
-y, --yes         If specified, will bypass all confirmation prompts. (default:
                  False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl repo component reload DUMMY help'

options:

scsctl repo thirdparty

Manages third-party software installations within the SCS repository.

Usage

```
usage: scsctl repo thirdparty [-v | --info-log | --debug-log | --trace-log]
                               [--user USER | --token TOKEN] [-c]
                               [-s {name,bytes,md5}] [--pretty]
                               options: ...
```

Manages third-party software installations within the Platform repository.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                     included, then this must be in the form
                     "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-c, --csv            If present, will list in CSV format. (default: False)
-s {name,bytes,md5}, --sort-by {name,bytes,md5}
                     The column to use for sorting. (default: name)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                     output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                     None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                     (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                     (default: None)
--pretty             If specified, will format output to be more human-
                     friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl repo thirdparty add help'

options:

```
add                 Adds third-party software to the Platform repository.
delete              Removes third-party software from the Platform
                    repository.
list                Lists installed third-party software within the
                    Platform repository.
```

scsctl repo thirdparty add

Adds third-party software to the SCS repository.

Usage

```
usage: scsctl repo thirdparty add
        [-v | --info-log | --debug-log | --trace-log]
        [--user USER | --token TOKEN]
        (-f FILE | -u URL) [--force]
options: ... name
```

Adds third-party software to the Platform repository.

positional arguments:

name The name that the repository should use for the uploaded software.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)

--token TOKEN Authentication token. (default: None)

-f FILE, --file FILE Path to software file to be added to the repository, or '-' to read from stdin. (default: None)

-u URL, --url URL URL of software file to be added to the repository. (default: None)

--force If set, forces overwrite of an existing matching repository entry. (default: False)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)

--info-log Set info-level verbosity (equivalent to -v). (default: None)

--debug-log Set debug-level verbosity (equivalent to -vv). (default: None)

--trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl repo thirdparty add DUMMY help'

options:

scsctl repo third party delete

Removes third-party software from the SCS repository.

Usage

```
usage: scsctl repo thirdparty delete
      [-v | --info-log | --debug-log | --trace-log]
      [--user USER | --token TOKEN] [-y]
      options: ... name
```

Removes third-party software from the Platform repository.

positional arguments:

name The name of the third-party software to be removed.

optional arguments:

--user USER User name and (optionally) password. If password is included, then this must be in the form "{user_name}:{password}" (default: None)
 --token TOKEN Authentication token. (default: None)

output options:

These options affect the way output is displayed.

-v, --verbose Increase output verbosity (ex: -v, -vv, -vvv). Verbose output is always sent to stdout. (default: None)
 --info-log Set info-level verbosity (equivalent to -v). (default: None)
 --debug-log Set debug-level verbosity (equivalent to -vv). (default: None)
 --trace-log Set trace-level verbosity (equivalent to -vvv). (default: None)
 -y, --yes If specified, will bypass all confirmation prompts. (default: False)

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl repo thirdparty delete DUMMY help'

options:

scsctl repo third party list

Lists installed third-party software within the SCS repository.

Usage

```
usage: scsctl repo thirdparty list
      [-v | --info-log | --debug-log | --trace-log]
      [--user USER | --token TOKEN] [-c]
      [-s {name,bytes,md5}] [--pretty]
      options: ...
```

Lists installed third-party software within the Platform repository.

optional arguments:

```
--user USER          User name and (optionally) password. If password is
                    included, then this must be in the form
                    "{user_name}:{password}" (default: None)
--token TOKEN        Authentication token. (default: None)
-c, --csv            If present, will list in CSV format. (default: False)
-s {name,bytes,md5}, --sort-by {name,bytes,md5}
                    The column to use for sorting. (default: name)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose        Increase output verbosity (ex: -v, -vv, -vvv). Verbose
                    output is always sent to stdout. (default: None)
--info-log           Set info-level verbosity (equivalent to -v). (default:
                    None)
--debug-log          Set debug-level verbosity (equivalent to -vv).
                    (default: None)
--trace-log          Set trace-level verbosity (equivalent to -vvv).
                    (default: None)
--pretty             If specified, will format output to be more human-
                    friendly. (default: False)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl repo thirdparty list DUMMY help'

options:

scsctl system

Performs administrative operations on the system.

Usage

```
usage: scsctl system [-v | --info-log | --debug-log | --trace-log]
                  [--user USER | --token TOKEN]
                  options: ...
```

Performs administrative operations on the system.

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN     Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose     Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log        Set info-level verbosity (equivalent to -v). (default: None)
--debug-log       Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log       Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl system reset help'

options:

```
reset            DANGER! This action will completely reset all post-
                  installation changes to your Platform server!
```

scsctl system reset

DANGER!

This action completely resets all post-installation changes to the SCS server!

Usage

```
usage: scsctl system reset [-v | --info-log | --debug-log | --trace-log]
                          [--user USER | --token TOKEN]
                          options: ...
```

DANGER! This action will completely reset all post-installation changes to your Platform server!

optional arguments:

```
--user USER      User name and (optionally) password. If password is included,
                  then this must be in the form "{user_name}:{password}"
                  (default: None)
--token TOKEN     Authentication token. (default: None)
```

output options:

These options affect the way output is displayed.

```
-v, --verbose     Increase output verbosity (ex: -v, -vv, -vvv). Verbose output
                  is always sent to stdout. (default: None)
--info-log        Set info-level verbosity (equivalent to -v). (default: None)
--debug-log       Set debug-level verbosity (equivalent to -vv). (default:
                  None)
--trace-log       Set trace-level verbosity (equivalent to -vvv). (default:
                  None)
```

subcommands:

Which action you wish to perform. To get detailed help, add 'help' to the end. For example: 'scsctl system reset DUMMY help'

options:

Content Application Development

See [Content Gateway Concepts](#) before you begin.

This section guides those in the following roles in developing applications that work with the Content Gateway:

- Storage system administrators
- End-user developers

The administrators are normally responsible for allocating storage, managing capacity, monitoring storage system health, replacing malfunctioning hardware, and adding additional capacity when needed. This can also include development staff responsible for automating storage administration functions.

End-user developers are responsible for creating custom application or integrating existing applications to use Content Gateway storage.

- [Gateway Metadata Transformation](#)
- [Migrating Applications from Direct-to-Swarm](#)
- [Metadata Translation between SCSP and S3](#)
- [Token-Based Authentication](#)
- [Restricting Domain Access](#)
- [Gateway Audit Logging](#)
- [Content Management API](#)
- [Content SCSP Extensions](#)

Gateway Metadata Transformation

- [Metadata Values](#)
 - [Metadata Substitution Variables](#)
- [XFORM Document Format](#)
 - [Example XFORM Document](#)

The metadata transformation facility allows domain and bucket administrators to define rules to add or replace metadata on incoming objects. These rules are stored in XFORM documents.

As a write operation (PUT, POST, or COPY) passes through the Gateway, rules in the **xform** sub-resource for a domain and/or bucket are applied and the object's headers is modified accordingly. Metadata rules specify a given header is added to the message if it does not exist or is replaced if it does exist in the request. Headers defined in the domain XFORM take precedence over anything defined in a bucket XFORM.

Metadata Values

Metadata values are specified as strings, with a small number of variables available for substitution, using a `${varname}` format where "varname" is the name of the variable.

Specify as a header value `${user}'s stuff` on the user rooster's bucket and objects written into that bucket end up with a header value of `rooster's stuff`.

Metadata Substitution Variables

Variable Name	Description
<code>date:format</code>	Create/update time stamp where <i>format</i> is defined by Java SimpleDateFormat specification
<code>user</code>	Authenticated user ID
<code>domain</code>	Domain name
<code>bucket</code>	Bucket name

XFORM Document Format

```
required Root := dict (
optional "metadata" := MetadataXforms
optional "comments" := any object type )
MetadataXforms := dict ( HttpHeaders,HttpHeaderValue )
HttpHeaderName := str # Conforms to HTTP spec
HttpHeaderValue := str # Conforms to HTTP spec, plus variables
```

Example XFORM Document

```
{
  "comments": "Metadata transform document",
  "metadata": {
    "X-Written-When-Meta": "${date:yyyyMMdd-HHmms}",
    "X-Contains-Meta": "${domain}/${bucket}",
    "X-Copyright-Meta": "Copyright ${date:yyyy}, MetaCorp, Inc",
    "X-Author-Meta": "${user}"
  }
}
```

Migrating Applications from Direct-to-Swarm

This section describes how to adapt native Swarm storage applications to use Content Gateway.

- [Requirements](#)
 - [Tip](#)
- [Domains](#)
- [Authentication](#)
 - [Deprecated](#)
- [SSL](#)

Requirements

- Supply storage domain name in all requests
- Use HTTP basic authentication instead of digest
- Use Gateway ACL system instead of native Swarm auth/auth
- Do not use Integrity Seal hash-type upgrade through Gateway



Tip

When integrating with Gateway, applications do not need to handle the HTTP 100-continue or redirect semantics Swarm clients must include: the Gateway operates as a reverse proxy and correctly use 100-continue when communicating with Swarm and hides all redirects from the upstream client.

Domains

Because Gateway is performing access control and validation for all operations, every content request must identify the storage domain for which the request is destined. The order of precedence for specifying the storage domain is:

1. Query argument: **domain=X**, else
2. HTTP **X-Forwarded-Host** header, else
3. HTTP request **Host** header value.

While some native integrations with Swarm are rigorous in specifying the storage domain, Swarm is permissive of requests not specifying one. Swarm also has additional precedence rules for assigning the storage domain; these are not compatible with requests handled through Gateway. When using Gateway, an application must specify the storage domain explicitly using one of the listed methods.

Authentication

It is common to require client applications to authenticate requests because Gateway is often deployed in access-controlled environments. While applications that previously integrated with Swarm may not have chosen to include provisions for authenticating requests.



Deprecated

The native Swarm auth/auth feature is deprecated and was removed as of June 2017. `security.noauth = False` must be added to continue using the native auth/auth if using Swarm's native auth/auth for applications.

Applications can interoperate with Gateway and Swarm by implementing the Gateway ACL system or using a library that provides for an automatic selection. Unless an application manipulates the access control policies within Swarm, no additional changes are required when integrating with Gateway. Applications that do manipulate these policies need to be adapted for Gateway's enhanced access control mechanism.

SSL

Content Gateway provides system administrators with the capability of encrypting client communications with SSL. Applications are recommended to provide for HTTPS communications when integrating with Gateway. Since many HTTP libraries already provide this capability, it is likely that applications need to add a configuration provision to use HTTPS versus HTTP.

Metadata Translation between SCSP and S3

Gateway performs translations of custom metadata formatting between the S3 and SCSP protocols as of release 5.4. Gateway now provides S3 and SCSP applications the ability to access each other's metadata.

Gateway provides the following translations for [Custom Metadata Headers](#) to allow SCSP and S3 clients to manipulate the full set of metadata Swarm Storage supports:

SCSP		S3
x-*-*meta	<- both ways ->	x-amz-meta-\1
x-*-*meta-*	<- both ways ->	x-amz-meta-\1-meta-\2
x-amz-meta-*	one way only ->	x-amz-meta-\1

S3 Client Issues

In an S3 client (e.g. Cyberduck object Info => Metadata tab) add two custom metadata values ("meta = cyberduck1" and "amz-meta = cyberduck2") as follows:

- x-amz-meta-meta: cyberduck1
- x-amz-meta-amz-meta: cyberduck2

Gateway merges to the same header name because occurrences of "amz-meta-" are removed when stored in Swarm. Here is how that custom metadata is returned in SCSP and S3:

SCSP HEAD

```
curl --head -u caringoadmin:password 'http://mydomain.example.com:9984/mybucket/duck.mpeg'
...
x-meta-meta: cyberduck1
x-meta-meta: cyberduck2
...
```

S3 HEAD

```
curl --head -u caringoadmin:password 'http://mydomain.example.com:9985/mybucket/duck.mpeg'
...
x-amz-meta-meta: cyberduck1
x-amz-meta-meta: cyberduck2
...
```

Note: Cyberduck uses the *last* header returned.

Token-Based Authentication

- [Note](#)

Content Gateway allows for the use of an optional token-based authentication in addition to HTTP Basic authentication. Token-based authentication works by performing a one-time HTTP Basic authentication request within the Management API or to a special URI path in the Storage API to receive a token. This token is used on subsequent requests as proof of the user's credentials.

Tokens have the following characteristics:

- They are always owned by the creating user except for tokens created by token administrators.
- They expire at a fixed time after creation; default is 24 hours if not specified.
- They may contain an optional S3 secret access key for use with the S3 protocol.
- They may contain optional metadata matching the prefix pattern: `x-custom-meta-*`
- The owner can list and delete active tokens.
- The token administrators can list and delete any user's active tokens.

Application developers may prefer to make use of the Management API to create tenant tokens for storage domains belonging to a tenant. Storage domain tokens are created with the special URI defined by the `tokenPath` IDSYS attribute.

The following is an example excerpt from a root IDSYS configuration file defining the token settings. Both the `cookieName` and `tokenPath` parameters must be defined to enable token-based authentication.

```
{
"ldap" : { ...
"cookieName": "token",
"tokenPath": "/.TOKEN/",
"tokenAdmin": "superuser@admindomain.example.com"
}
}
```

Tokens are delivered using the standard HTTP cookie mechanism. The `cookieName` parameter is the cookie's name and the value is the token. The token value is guaranteed to be universally unique and impossible to guess. The `tokenPath` parameter defines the URI path within the storage domain with which a user requests a token and then performs listing and delete operations on active tokens. The `tokenAdmin` is the user name of the token administrator able to create, list, and delete tokens on behalf of other users.

The token administrator is recommended to be a fully qualified user name to avoid ambiguity in a situation where a storage domain may inherit the IDSYS from the tenant or root scope.

See *"Qualification of User/Group Names" in the [IDSYS Document Format](#).*

Gateway stores all tokens within the administrative domain as automatically expiring objects using the object lifepoint feature. The expiration time of an authentication token can be specified when the token is created. A default expiration time is assigned based on the `tokenTTLHours` parameter in the `[gateway]` section of the `gateway.cfg` file if the time is not specified. The request proceeds as an anonymous user subject to all normal access control policies if an expired token is presented to Gateway. The `Set-Cookie` header of the response instructs the HTTP client to delete the expired token cookie.

POST a blank document to either of the following to create a new authentication token:

- the storage domain and token path, or
- the Management API path `/_admin/manage/tenants/{tenantName}/tokens/`

Use HTTP Basic authentication to authenticate the request. Requests to the `tokenPath` URI are processed independently from the storage protocol handling and these instructions work with both SCSP and S3 front-end protocols and to the Management API.

**Note**

HTTP Basic authentication is demonstrated using "Auth: {user}:{password}" for clarity. Use the `Authorization` HTTP request header according the definition in [RFC 2717](#).

Managing Tokens

- [Listing Authentication Tokens](#)
- [Removing an Authentication Token](#)
- [Clearing Tokens for Locked Accounts](#)
- [Token Examples](#)

Listing Authentication Tokens

Perform a GET on the token path using an existing authentication token or using HTTP basic authentication to validate the request to list active authentication tokens.

Listing domain tokens

```
GET http://{domain}/.TOKEN/?format=json
Cookie: token=d9f8378f71e79b77831f65d9e6891af6

HTTP/1.1 200 OK
Gateway-Request-Id: F48303758301E570
Castor-Object-Count: 3
Content-Type: application/json; charset=utf-8
Content-Length: 651
[
  {
    "x_token_domain_meta": "{domain}",
    "x_owner_meta": "john",
    "last_modified": "2012-06-22T05:39:44.854100Z",
    "lifepoint": "[Sat, 23 Jun 2012 05:39:44 GMT] reps=2,[] delete",
    "name": "7e742e12fb7e070b44266df1a1bf2efe"},
    ...
  ]
```

Listing tenant tokens

```
GET http://{domain}/_admin/manage/tenants/tenant256/tokens/
Authorization: Basic Z2NhcmxpbjpmYW5ueQ==
```

Removing an Authentication Token

Perform a DELETE on the full token path and authenticate the request with a token or with HTTP basic authentication to logout and remove an authentication token.

Deleting a domain token

```
DELETE http://{domain}/.TOKEN/53dfb96dc6d5b9cacd174e3649cba6d5
Cookie: token=22f57e203c10cf86d2dfd9564b1413f5
```

Deleting a tenant token

```
DELETE http://{domain}/_admin/manage/tenants/tenant256/tokens/53dfb96dc6d5b9cacd174e3649cba6d5
Authorization: Basic Z2NhcmxpbjpmYW5ueQ==
```

The Gateway returns a `Set-Cookie` header to clear the token if a token is deleted and the same deleted token is used to authenticate the request. This is useful when implementing logout pages for web browsers.

Deleting a domain token with itself

```
DELETE http://{domain}/.TOKEN/53dfb96dc6d5b9cacd174e3649cba6d5
Cookie: token=53dfb96dc6d5b9cacd174e3649cba6d5

HTTP/1.1 200 OK
Gateway-Request-Id: 9855371AA8411781
Set-Cookie: token=; path=/
Content-Length: 0
```



Note

The operation must be authenticated using either the token within a **Cookie** header or by using a valid user and password in an **Authentication** header with the request when using the token in the URI path. The audit log reflects the name of user that owns the token if the cookie is used or the name of the authenticated user if HTTP basic authentication is used.

Clearing Tokens for Locked Accounts

Gateway allows unexpired tokens to continue to work for locked accounts because identity management systems are poor at signalling that an account has been locked. The token stops working as soon as it expires from cache for a *removed* account.

Extra measures are needed to verify the tokens stop working for an account that is expired (locked) but *not removed*:

PAM Authentication:

This method is for those using a PAM as a front-end for traditional Unix authentication.

1. Lock the user account by change the password: `passwd -l USERNAME`
2. Change the username: `zzzUSERNAME`

LDAP Authentication:

1. Standardize an attribute within one of the schemas that apply to the user record for which enabled user accounts always have set to a known value.
2. Design a test for the value.



Tip

Although negative test can be used to find disabled accounts, there is less risk of mistakes with the affirmative method: attribute is value.

Use the `pwdPolicy` schema with the `pwdLockout` attribute and use the `userFilter` to require the `pwdLockout` attribute to be true.

Token Examples

The token administrator defined in the root `IDSYS` configuration file is allowed to use the `x-owner-meta` argument to perform token listing for any user. Administrators wishing to disable a user account and log them out of the system can do so by locking the LDAP account and then removing any existing authentication tokens for the user.

The following examples show how the token administrator lists and deletes another user's tokens.

Token administrator `superuser@admindomain.example.com` listing the authentication tokens for user `john`:

Discovering tokens

```
GET http://{domain}/.TOKEN/?format=json&x-owner-meta=john
Auth: superuser@admindomain.example.com:superpassword

HTTP/1.1 200 OK
Gateway-Request-Id: 29172D0FDCAB19DE
Castor-Object-Count: 1
Content-Type: application/json; charset=utf-8
Content-Length: 221
[
  { "x_token_domain_meta": "{domain}",
    "x_owner_meta": "john",
    "last_modified": "2012-06-24T07:14:53.671600Z",
    "lifepoint": "[Mon, 25 Jun 2012 07:14:53 GMT] reps=2,[] delete",
    "name": "b71805b6c862860bfed892c653cbc4b5" }
]
```

Using the tokens discovered during the listing operation, the token administrator then issues deletes for each of the tokens in exactly the same way the user deletes tokens.

Deleting token

```
DELETE http://{domain}/.TOKEN/b71805b6c862860bfed892c653cbc4b5
Auth: superuser@admindomain.example.com:superpassword

HTTP/1.1 200 OK
Gateway-Request-Id: 4628361DE8318726
Content-Length: 0
```

Notice the token administrator lists tokens the same way any user does and is able to specify an arbitrary user with the `x-owner-meta` query argument. The delete operation is the same pattern whether performed by the user or the token administrator.



Best practice

Use the token administrator's credentials when accessing or deleting tokens for other users so the audit log reflects the token administrator performed the operations.

Creating Tokens

- [Query Arguments for Tokens](#)
 - [Note](#)
- [Request Headers for Tokens](#)
- [Token Examples](#)
 - [Creating a domain token](#)
 - [Creating a tenant token for S3](#)
 - [Creating a tenant token for S3 with cURL](#)

Query Arguments for Tokens

The following HTTP request URI query arguments control the creation of a token:

No query args	Causes the default behavior as if <code>setcookie=true</code> was specified.
setcookie=true	Causes the HTTP response to contain a Cookie header that causes a web browser to replace the current authentication token with the newly generated one.
setcookie=false	Causes the HTTP response to contain the header Gateway-Token instead of the standard Cookie header. Use this to have the browser continue using its current authentication token.

 **Note**

The Gateway-Token header is the same for both SCSP and S3 tokens.

Request Headers for Tokens

The following HTTP request headers control the creation of a token:

X-Owner-Meta	{username}	Required	Used by the <code>tokenAdmin</code> user to create a token on behalf of another user. An error is returned if any user other than the token administrator attempts to set this header. By default, the owner of a token is the user that creates it.
X-User-Token-Expires-Meta	{time-specification}	Optional	Sets the expiration time for the authentication token. See below for ways to express time. If this header is not given, the default expiration time is set based on Gateway's <code>tokenTTLHours</code> configuration setting, which defaults to 24 hours after token creation.
X-User-Secret-Key-Meta	{string}	Optional	Sets an S3 secret key that is used for signing S3 requests. The token is used to sign S3 storage requests when this header is present. The token cannot be used to authenticate SCSP storage or Management API operations. Values of this string must follow Swarm metadata value rules for encoding, and 7-bit ASCII values are recommended.

X-Custom-Meta- {string}	{string}	Optional	Additional custom metadata that is saved with the token. This is for application-specific purposes and it is not interpreted by the Gateway during token creation or use.
X-Custom-Meta-Source	{string}	Optional	This metadata header is displayed as the Description of the token in the Content UI.

You have numerous options for how to specify the time for the token's expiration:

POSIX time	{n}	"1444419929"	Integer value that is the number of seconds elapsed since 00:00:00. Coordinated Universal Time (UTC), 1 January 1970, not counting leap seconds.
Days offset	+(n)	"+365"	Integer number of days (86,400 sec/day) from now.
Year only	{YYYY}	"2015"	Four-digit year; the expiration is on January 1st at 00:00Z of that year.
Specific day	{YYYY}{MM}{DD}	"2015-10-09"	Year, month, and day; the expiration is at 00:00Z on that day.
ISO timespec	{YYYY}{MM}{DD} T{hh}:{mm}:{ss}.{nnn} Z	"2015-10-09T11:18:00.000Z"	ISO time specification; all digits and fixed characters must be supplied; only UTC ("Z") time zone is allowed.

Token Examples

Creating a domain token

```
POST http://{domain}/.TOKEN/
Auth: john:password
```

```
HTTP/1.1 201 Created
Gateway-Request-Id: 41B8FD0D739DF86C
Set-Cookie: token=d9f8378f71e79b77831f65d9e6891af6; path=/
Content-Length: 0
```

Creating a tenant token for S3

```
POST http://{domain}/_admin/manage/tenants/tenant256/tokens/
Auth: john:password
X-User-Token-Expires-Meta: +730
X-User-Secret-Key-Meta: 5ZdMSEubcFHJjnkyEzy722ZQHjd2xsTo
X-Custom-Meta-Source: Laptop Applications
```

```
HTTP/1.1 201 Created
Gateway-Request-Id: 7612F7FDB63B7C02
Set-Cookie: token=cc8ea2467d196b047497818f6271f00c; path=/
Content-Length: 0
```

Creating a tenant token for S3 with cURL

```
$ USER="john"
$ SECRETKEY="1NnYIOXeHfuuW30eARH19iJQXNvvjMSF"
$ EXPIRES="+365"
$ curl -u $USER -X POST --data-binary "" \
-H "X-User-Secret-Key-Meta: $SECRETKEY" \
-H "X-User-Token-Expires-Meta: $EXPIRES" \
"http://mydomain.example.com/_admin/manage/tenants/tenant255/tokens/"
Enter host password for user 'john':
{"token": "8c3955185d3ae8347cacal14e4e2416", ... }
```

Restricting Domain Access

Cluster administrators inevitably need to cut off some or all access to the hosted domains within a cluster when Gateway is deployed by a managed service provider. This can be due to non-payment or if a client uses too much storage and is required to clean-up space before writing new content.

All access to a domain can be controlled from the root Policy configuration file and from the domain's `policy` attribute. Updating the `policy` attribute is often desirable because, unlike an update to the root Policy file, it does not require a Gateway server restart. These examples use the `policy` attribute of a domain for controlling access. Recall the statements in an access Policy have an optional `Sid` field that can be used in whatever way an application wants. Administrators can use the `Sid` field to keep track of the statements they added and to identify them for future removal when injecting statements into an existing Policy.

- [No Access](#)
- [Read-Only Access](#)
- [Read- and Delete-Only Access](#)

No Access

In this example, a domain that had allowed access to the domain administrator (one of the end-users) now completely cuts off access to all end-users by adding the deny statements. The new statements use the **Sid** field to identify them for easy removal in the future. Notice the statement denies authenticated users as well as anonymous users.

```
{
  "Statement": [
    {
      "Resource": "/*",
      "Action": [
        "*"
      ],
      "Principal": {
        "user": [
          "domainadmin"
        ]
      },
      "Effect": "Allow"
    },
    {
      "Resource": "/*",
      "Action": [
        "*"
      ],
      "Principal": {
        "user": [
          ""
        ],
        "anonymous": [
          ""
        ]
      },
      "Effect": "Deny",
      "Sid": "temp-cutoff-noaccess"
    }
  ]
}
```

Read-Only Access

In this example, a domain is changed to read-only mode to prevent writing, updating, or deleting content by the-end users. The new policy statement makes use of the **Sid** field to identify it for future removal. This example also makes use of **NotAction** to specify the deny pertains to any action not listed thus allowing the ones that are listed.

```
{
  "Statement": [
    {
      "Resource": "/*",
      "Action": [
        "*"
      ],
      "Principal": {
        "user": [
          "domainadmin"
        ]
      },
      "Effect": "Allow"
    },
    {
      "Resource": "/*",
      "NotAction": [
        "GetObject",
        "GetBucket",
        "GetDomain",
        "ListBucket",
        "ListDomain",
        "GetDomainPolicy",
        "GetPolicy",
        "PutPolicy"
      ],
      "Principal": {
        "user": [
          ""
        ],
        "anonymous": [
          ""
        ]
      },
      "Effect": "Deny",
      "Sid": "temp-cutoff-ro"
    }
  ]
}
```

Read- and Delete-Only Access

A cluster administrator can set the access control policy on a domain for read and delete only if a tenant exceeds the quota. By letting the end-users continue to read and delete content, they can use the content they have already written and clean-up content to reduce storage usage. As with the previous example, **NotAction** is used to specify the deny pertains to any action not listed.

```
{
  "Statement": [
    {
      "Resource": "/*",
      "Action": [
        "*"
      ],
      "Principal": {
        "user": [
          "domainadmin"
        ]
      },
      "Effect": "Allow"
    },
    {
      "Resource": "/*",
      "NotAction": [
        "GetObject",
        "GetBucket",
        "GetDomain",
        "ListBucket",
        "ListDomain",
        "GetDomainPolicy",
        "GetPolicy",
        "PutPolicy",
        "DeleteObject",
        "DeleteBucket",
        "DeleteDomain"
      ],
      "Principal": {
        "user": [
          ""
        ],
        "anonymous": [
          ""
        ]
      },
      "Effect": "Deny",
      "Sid": "temp-cutoff-readdelete"
    }
  ]
}
```

Gateway Audit Logging

Gateway's audit log of user actions is designed for machine parsing so it can be used for auditing, compliance monitoring, API request analysis, and SLA reporting.

See [Gateway Configuration](#) for configuring the logging output.

- [Audit Log Message Fields](#)
- [Audit Log Message Formats](#)
- [Example Log Messages](#)
- [Behaviors of Operations](#)
- [Application-Supplied Tag](#)

Audit Log Message Fields

This section focuses on the format of the audit logs to allow for integration and development of applications that use them.

These are the fields that appear in logging output. These are only definitions and not the format of any particular log message.

Field Name	Description
Auth Domain	Tenant or storage domain name used to authenticate user; tenant names prefixed with "+"
Auth User	User ID used to authenticate; empty if anonymous
Bucket	Name of bucket
DNS Domain	Origin DNS domain; value of Host header from the request
Domain	Swarm domain name to which operation refers to
Elapsed Time	Transaction time in milliseconds
HTTP Code	Request response code. Exceptions in request handling return a 500. All SCSP requests with authorization errors output a 401.
Log Level	Logging level for the audit log entry
Message Type	Message category to simplify filtering
Object Name or UUID	Named of object, excluding bucket name, or UUID for unnamed streams
Operation	The operation. Examples: POST, HEAD, DELETE, INVOKE

Record Format Version	Audit log record format version. This changes if format of the output records is different from the previous release.
Request ID	A unique identifier for client request attached to all associated audit messages. This value matches the HTTP response header Gateway-Request-Id given to the client and is used in the server log.
Response Bytes	Number of bytes sent to Source IP in the HTTP response body
Source Bytes	Number of bytes received from Source IP in the message body
Source IP	IP address from which a request originated
Timestamp	High resolution timestamp up to millisecond

Audit Log Message Formats

Following are the output formats for all event types. All log messages share a common set of prefix fields, which includes a message type. The suffix fields in a log message are variable based on the message type. This allows for automated parsing of log messages.

The fields in each log message are separated by spaces. If a field value is missing, the string (none) is substituted. Field values are subject to HTML URL encoding to make spaces, UTF-8, and other special characters safe for inclusion in the audit log entry.

- Alphanumeric characters "a" through "z", "A" through "Z" and "0" through "9" remain unchanged
- Characters ".", "-", "*", and "_" remain unchanged
- Space character is converted into a plus sign "+"
- All other characters are converted into %HH byte values using UTF-8 encoding



Note

The "/" character in an object's name appear as "%2F" in the log, based on the previous rules.

Common Prefix Fields

All messages are prefixed by the following fields in this order:

1. Timestamp
2. Log Level
3. Request ID
4. Record Format Version
5. Source IP
6. DNS Domain
7. Message Type
8. Operation
9. Auth User
10. Auth Domain
11. HTTP Code
12. Source Bytes

- 13. Response Bytes
- 14. Elapsed Time

Suffix Fields

This table defines the suffix fields that are included with each log message following the common prefix fields.

Event	Message Type	Operation	Suffix Fields		
User requests token	Auth	GET			
User deletes token		DELETE			
List available domains	Admin	LIST_DOMAINS			
Domain creation	Domain	POST	Domain		
Domain policy create/ update		POLICY_PUT			
Domain policy read		POLICY_GET			
Domain policy delete		POLICY_DELETE			
Domain copy		COPY			
Domain delete		DELETE			
Domain read		GET			
Domain info		HEAD			
List buckets in a domain		LIST_BUCKETS			
Bucket creation		Bucket		POST	Domain, Bucket
Bucket policy create/ update				POLICY_PUT	
Bucket policy read	POLICY_GET				
Bucket policy delete	POLICY_DELETE				
Bucket copy	COPY				
Bucket delete	DELETE				
Bucket read	GET				
Bucket info	HEAD				
List objects in a bucket	LIST_OBJECTS				
S3 list multipart	LIST_MULTIPARTS				
Object creation	Scsp		POST	Domain, Bucket, Object name or UUID	
Object update		PUT			
Object append		APPEND			
Object copy		COPY			
Object delete		DELETE			
Object read		GET			

Object info	HEAD	
S3 multipart initiate	MULTIPART_INITIATE	Domain, Bucket, Object name
S3 multipart put	MULTIPART_PUT	
S3 multipart copy	MULTIPART_COPY	
S3 multipart abort	MULTIPART_ABORT	
S3 multipart complete	MULTIPART_COMPLETE	
S3 list multipart	LIST_MULTIPART	

Example Log Messages

These are examples of a variety of audit log messages.

Successful login for user muser1 to the domain nom.dom.com

```
2019-05-13 19:28:29,671 INFO [9D9A577B66D2DD56] 2 172.20.1.1 172.20.1.2
Auth POST muser1 nom.dom.com 201 0 0 0.48
```

Successful POST of a bucket named redbucket by user admin1

```
2019-05-13 19:28:25,070 INFO [7169E3D6DD5656B9] 2 172.20.1.1 172.20.1.2
Bucket POST admin1 nom.dom.com 201 0 44 0.65 nom.dom.com redbucket
```

401 authentication challenge on a HEAD to an unauthenticated request

```
2019-05-13 19:28:36,632 INFO [85822E93CFBC6F12] 2 172.20.1.1 172.20.1.2
Bucket HEAD (none) nom.dom.com 401 0 0 0.72 nom.dom.com redbucket
```

Writing an object named water.jpg to bucket bluebucket without being required to authenticate

```
2019-05-15 14:54:31,616 INFO [D2AC19A94ECA5A51] 2 172.20.1.1 172.20.1.2
Scsp POST (none) open.dom.com 201 10 44 1.05 open.dom.com bluebucket water.jpg
```

Reading an object named water.jpg to bucket bluebucket without being required to authenticate

```
2019-05-15 14:54:31,818 INFO [86B6E646C65DC83B] 2 172.20.1.1 172.20.1.2
Scsp GET (none) open.dom.com 200 0 10 1.12 open.dom.com bluebucket water.jpg
```

Listing a bucket without being required to authenticate

```
2019-05-15 14:54:45,236 INFO [C87A09C1FCCCE581] 2 172.20.1.1 172.20.1.2
Bucket LIST_OBJECTS (none) open.dom.com 200 0 273 2.57 open.dom.com bluebucket
```

Listing a domain as user admin1

```
2019-05-15 16:32:14,560 INFO [CAE97BE991DE877A] 2 172.20.1.1 172.20.1.2
Domain LIST_BUCKETS admin1 nom.dom.com 200 0 180 2.38 nom.dom.com
```

Administrative override and replacement of domain's Policy by user superuser from root IDSYS

```
2019-10-16 10:37:29,719 INFO [D580617E135E35DF] 2 172.30.1.1 172.20.1.2
Domain POLICY_PUT !superuser@ nom.dom.com 201 123 0 1.08 nom.dom.com
```

Behaviors of Operations

Interrupted GET – When a GET operation is interrupted, such as if the socket closed unexpectedly prior to reading all data, the audit log may record an HTTP 200 response with response bytes equal to the size of the object. When interruption takes place, an HTTP 500 response is logged with response bytes equal to the actual number of bytes transmitted.

Duplicate Request IDs – All messages have the same Request ID so they can be correlated with the client request if multiple messages are logged from one client operation. For example, the recursive delete operation generates synthetic delete requests all with the same Request ID.

INVOKE operations – The optional feature *Video Clipping* (v11.0) logs INVOKE operations. Each video clipping event logs multiple events to provide auditing through the process, which may take a while to complete. When you create a video clip, Gateway acknowledges the request with an INVOKE message. See [Video Clipping for Partial File Restore](#).

Application-Supplied Tag

Gateway's audit logging allows for the client application to supply a custom tag that can be used to correlate multiple audit log entries to one application-level transaction. The application specifies this tag in a Gateway-Audit-Id request header and it must be alpha-numeric and is truncated at 32 characters. When this optional tag is received, the Request ID field of the audit log entry contains the automatically-generated request identifier from the Gateway, a dash ("-"), and the application-supplied tag.

Example of a normal request identifier and one with the application supplied tag trans123

```
2019-12-10 09:30:45,360 INFO [1813AC1764D48125] ...
2019-12-10 09:30:45,360 INFO [2AF5F226122D9673-trans123] ...
```

When the application-supplied tag is used for multiple operations, even across multiple Gateway servers, the request identifiers remain unique with a common suffix.

Content Management API

The Content Management API is an integration point for cloud management platforms and end-user applications. The Management API is purely administrative: it makes available actions for provisioning and managing storage tenants, domains, and other aspects of the Swarm cloud storage infrastructure using the same authentication and access control policy mechanism used within the Storage API.

The Management API is implemented as an HTTP/1.1 REST interface that is separate from the SCSP and S3 storage interfaces and is available for every user that can access the system. It works by overlaying the storage API name space at the `/_admin/manage/` URI prefix.

Note
 Since the `_admin` bucket is already a reserved resource for use by Swarm only, this name space overlay should have no effect on existing end-user applications.

- [Other Gateway Requests](#)
- [Request Methods for Tenants](#)
- [Request Methods for Storage Domains](#)
- [Management API Response Formats](#)
- [Namespace Structure](#)
- [Request Methods for Buckets](#)
- [Defined ETC Documents](#)
- [Methods for Quotas](#)
- [Domain Adoption](#)

Other Gateway Requests

This is detailed information about the request methods for other Content Management API resources.

- [Read API Version](#)
- [Read Cluster Storage Usage](#)

Read API Version

Method	GET
URI Suffix	version
Query Args	
Headers	
Policy Action	
Description	Read the Management API version information
Restrictions	
Request Body	
Response	JSON version response

Read Cluster Storage Usage

Method	GET
URI Suffix	meter/cluster/usage
Query Args	
Headers	
Policy Action	ListTenants
Description	Read the storage usage information for the back-end cluster
Restrictions	
Request Body	
Response	JSON cluster usage response

Request Methods for Tenants

This is detailed information about the request methods for tenants. Using these methods with curl has this format (here, for listing):

```
curl -i -u caringoadmin:pwd https://site.example.com/_admin/manage/tenants/
```



Note

Tenant names are converted to lowercase before evaluation.

- [Note](#)
- [List Tenants](#)
- [Create Tenant](#)
- [Read Tenant](#)
- [Delete Tenant](#)
- [List Tenant ETC Documents](#)
- [Create Tenant ETC Document](#)
- [Read Tenant ETC Document](#)
- [Delete Tenant ETC Document](#)
- [List Authentication Tokens](#)
- [Create Authentication Token](#)
- [Read Authentication Token](#)
- [Delete Authentication Token](#)

See [Defined ETC Documents](#) for more on ETC documents (IDSYS, Policy, XFORM).

List Tenants

Method	GET
URI Suffix	tenants
Query Args	
Headers	
Policy Action	ListTenants
Description	Returns a list of tenants
Restrictions	Paging is not supported in the request; only 1000 returned at this time.
Request Body	
Response	JSON formatted tenant listing response

Create Tenant

Method	PUT
URI Suffix	tenants/{tenant}
Query Args	
Headers	Optional metadata to be saved with tenant
Policy Action	CreateTenant
Description	Create a tenant named {tenant}. If tenant already exists, this action overwrites the metadata for the tenant.
Restrictions	Tenant name must be 7-bit ASCII characters in the set [a-z, 0-9, hyphen].
Request Body	
Response	JSON general request response

Read Tenant

Method	GET
URI Suffix	tenants/{tenant}
Query Args	
Headers	
Policy Action	GetTenant
Description	Reads a tenant object and metadata
Restrictions	
Request Body	
Response	Tenant object body (normally null) and metadata

Delete Tenant

Method	DELETE
URI Suffix	tenants/{tenant}
Query Args	recursive=yes (required)
Headers	

Policy Action	DeleteTenant
Description	Deletes all data related to a tenant including storage domains
Restrictions	
Request Body	
Response	JSON general request response

List Tenant ETC Documents

Method	GET
URI Suffix	tenants/{tenant}/etc
Query Args	
Headers	
Policy Action	ListEtc
Description	Returns a list of tenant documents
Restrictions	Paging is not supported in the request; only 1000 returned at this time.
Request Body	
Response	JSON formatted documents listing response

Create Tenant ETC Document

Method	PUT
URI Suffix	tenants/{tenant}/etc/{document}
Query Args	
Headers	Any metadata to be included with the document
Policy Action	PutPolicy
Description	Create or overwrite a document associated with the tenant
Restrictions	Maximum document size is 1MB.
Request Body	The document contents
Response	JSON general request response

Read Tenant ETC Document

Method	GET
URI Suffix	tenants/{tenant}/etc/{document}
Query Args	
Headers	
Policy Action	GetPolicy
Description	Read a tenant document
Restrictions	
Request Body	
Response	Document body and metadata

Delete Tenant ETC Document

Method	DELETE
URI Suffix	tenants/{tenant}/etc/{document}
Query Args	
Headers	
Policy Action	DeletePolicy
Description	Delete a tenant document
Restrictions	
Request Body	
Response	JSON general request response

List Authentication Tokens

Method	GET
URI Suffix	tenants/{tenant}/tokens
Query Args	Set x-owner-meta={user} to search for another user's tokens; set withsecrets=true to include secret fields
Headers	

Policy Action	ListTokens
Description	List user authentication tokens
Restrictions	
Request Body	
Response	JSON formatted token listing response

Create Authentication Token

Method	POST
URI Suffix	tenants/{tenant}/tokens
Query Args	Set setcookie=true to have newly created token included in the Cookie response header. Set setcookie=false to prevent the newly created token from inclusion in a Cookie header in the response.
Headers	See Token-Based Authentication for details.
Policy Action	CreateToken
Description	Create user authentication token
Restrictions	Does not support creation of tokens for other users
Request Body	
Response	JSON formatted token response

Read Authentication Token

Method	GET
URI Suffix	tenants/{tenant}/tokens/{token}
Query Args	
Headers	
Policy Action	ValidateToken
Description	Read user authentication token
Restrictions	
Request Body	
Response	JSON formatted token response

Delete Authentication Token

Method	DELETE
URI Suffix	tenants/{tenant}/tokens/{token}
Query Args	
Headers	
Policy Action	DeleteToken
Description	Delete user authentication token
Restrictions	
Request Body	
Response	JSON general request response

Request Methods for Storage Domains

This is detailed information about the request methods for tenant storage domains. Using these methods with curl has this format (here, for listing):

```
curl -i -u caringoadmin:pwd https://site.example.com/_admin/manage/tenants/t1/domains/
```

Note

Storage domain names are converted to lowercase before evaluation.

- [Note](#)
- [List Tenant Domains](#)
- [Create Storage Domain](#)
- [Read Storage Domain](#)
- [Delete Storage Domain](#)
- [List Storage Domain ETC Documents](#)
- [Create Storage Domain ETC Document](#)
- [Read Storage Domain ETC Document](#)
- [Delete Storage Domain ETC Document](#)
- [Get Domain UUID by Name](#)
- [Get Domain Name by UUID](#)

See [Defined ETC Documents](#) for more on ETC documents (IDSYS, Policy, XFORM).

List Tenant Domains

Method	GET
URI Suffix	tenants/{tenant}/domains
Query Args	
Headers	
Policy Action	ListDomains
Description	Returns a list of storage domains owned by the tenant.
Restrictions	
Request Body	
Response	JSON formatted tenant listing response

Create Storage Domain

Method	PUT
URI Suffix	tenants/{tenant}/domains/{domain}
Query Args	
Headers	Optional metadata to be saved with domain
Policy Action	CreateDomain
Description	Create a domain named {domain}. If domain already exists, this action overwrites the metadata for the domain.
Restrictions	Domain name must be 7-bit ASCII characters in the set [a-z, 0-9, hyphen].
Request Body	
Response	JSON general request response

Read Storage Domain

Method	GET
URI Suffix	tenants/{tenant}/domains/{domain}
Query Args	
Headers	
Policy Action	GetDomain
Description	Reads a storage domain object and metadata. *Not available for the System domain
Restrictions	
Request Body	
Response	Domain object body (normally null) and metadata

Delete Storage Domain

Method	DELETE
URI Suffix	tenants/{tenant}/domains/{domain}
Query Args	recursive=yes (required)
Headers	

Policy Action	DeleteDomain
Description	Deletes all data related to the storage domain. *Not available for the System domain
Restrictions	
Request Body	
Response	JSON general request response

List Storage Domain ETC Documents

Method	GET
URI Suffix	tenants/{tenant}/domains/{domain}/etc
Query Args	
Headers	
Policy Action	ListEtc
Description	Returns a list of storage domain documents.
Restrictions	Paging is not supported in the request; only 1000 returned at this time.
Request Body	
Response	JSON formatted documents listing response

Create Storage Domain ETC Document

Method	PUT
URI Suffix	tenants/{tenant}/domains/{domain}/etc/{document}
Query Args	
Headers	Any metadata to be included with the document.
Policy Action	PutPolicy
Description	Create or overwrite a document associated with the storage domain. *Only the Policy ETC Document is supported for the System domain
Restrictions	Maximum document size is 1MB.
Request Body	The document contents
Response	JSON general request response

Read Storage Domain ETC Document

Method	GET
URI Suffix	tenants/{tenant}/domains/{domain}/etc/{document}
Query Args	
Headers	
Policy Action	GetPolicy
Description	Read a storage domain document. *Only the Policy ETC Document is supported for the System domain
Restrictions	
Request Body	
Response	Document body and metadata

Delete Storage Domain ETC Document

Method	DELETE
URI Suffix	tenants/{tenant}/domains/{domain}/etc/{document}
Query Args	
Headers	
Policy Action	DeletePolicy
Description	Delete a storage domain document
Restrictions	
Request Body	
Response	JSON general request response

Get Domain UUID by Name

Method	GET
URI Suffix	tenants/{tenant}/domains/{domain}/uuid
Query Args	
Headers	

Policy Action	GetDomain
Description	Returns the UUID (context ID) for a domain *Not available for the System domain
Restrictions	
Request Body	
Response	JSON general request response

Get Domain Name by UUID

Method	GET
URI Suffix	tenants/{tenant}/domains/{domainUUID}/name
Query Args	
Headers	
Policy Action	GetDomain
Description	Returns the canonical name for a domain *Not available for the System domain
Restrictions	
Request Body	
Response	JSON general request response

Management API Response Formats

These are the Content Management API response formats that can be returned with a request. The order of the JSON fields in output records and the ordering of lists is not defined unless otherwise noted.



Best practice

Make applications tolerant of extra fields in all responses for future compatibility.

- [General Request Response](#)
- [Version Response](#)
- [Cluster Usage Response](#)
- [Tenant, Domain Listing Response](#)
- [ETC Listing Response](#)
 - [Document Listing Response Format](#)
- [Token Response Formats](#)

General Request Response

This is the general-purpose response format for requests when no other specific format is defined. This response is also given when errors occur preventing the return of a specific response format. This general request response format is used to communicate the error condition if a listing operation fails.

The response is a JSON formatted body whose fields and HTTP status codes are defined as follows:

General Request Response Format

message	human-readable message describing response
code	text response code
errors	array of strings with error details

Response Code Text Strings

WriteSucceeded	201 status code
DeleteSucceeded	200 status code
CreateFailed	various status codes from storage cluster
ListFailed	various status codes from storage cluster
ReadFailed	various status codes from storage cluster
WriteFailed	various status codes from storage cluster
DeleteFailed	various status codes from storage cluster
NotAuthorized	401 or 403 status code
ServerError	500 status code
MissingParameter	400 status code; missing required query argument
NotFound	404 status code; message contains path
OtherError	various status codes; non-specific error
BadJson	400 status code; JSON unable to be parsed
InvalidJson	400 status code; JSON did not validate
BadRequest	400 status code; invalid argument or path

This is an example response:

```
{
  "message": "The data were not parsable JSON.",
  "code": "BadJson",
  "errors": [
    "Unexpected character ('P' (code 80)): expected a valid value
      (number, String, array, object, 'true', 'false' or 'null')\n at
      [Source: [B@37994099; line: 1, column: 2]"
  ]
}
```

Version Response

The version response contains information about the Management API software version. A general request response is given if an error occurs on the request.

manageApiVersion	API version string; format not defined
-------------------------	--

This is an example version response:

```
{  
  "manageApiVersion": "1.0"  
}
```

Cluster Usage Response

The cluster usage response contains storage usage information about the back-end Swarm object storage cluster. A general request response is given if an error occurs on the request.

availableGb	Storage GBytes available; integer
capacityGb	Total storage GBytes in the cluster; integer

This is an example version response:

```
{  
  "availableGb": 31676,  
  "capacityGb": 512437  
}
```

Tenant, Domain Listing Response

The tenant and domain listing response returns a listing of tenants and storage domains as a JSON list object. A general request response is given if an error occurs on the request.

name	Name of tenant or domain
etag	ETag opaque string; strong validator
contentMd5	base64 encoded MD5 value
lastModified	ISO 8601 format date and time with sub-second resolution

This is an example of a storage domain listing:

```
[
  {
    "name": "domain1.cloud.example.com",
    "etag": "74fd9321b793f0f62653e28f28e6e792",
    "contentMd5": "1B2M2Y8AsgTpgAmY7PhCfg==",
    "lastModified": "2013-11-24T20:06:07.719100Z"
  },
  {
    "name": "domain2.cloud.example.com",
    "etag": "237233ac1fddd2cd430920dcd133c3ac",
    "contentMd5": "1B2M2Y8AsgTpgAmY7PhCfg==",
    "lastModified": "2013-12-04T12:25:04.352100Z"
  },
  {
    "name": "domain3.cloud.example.com",
    "etag": "e224aa1d76ae7af7d77a94880f1a016e",
    "contentMd5": "1B2M2Y8AsgTpgAmY7PhCfg==",
    "lastModified": "2014-01-21T09:00:22.138100Z"
  }
]
```

ETC Listing Response

The etc listing response returns a listing of documents associated with a tenant or storage domain as a JSON list object. A general request response is given if an error occurs on the request.

Document Listing Response Format

name	Name of tenant or domain
etag	ETag opaque string; strong validator
contentMd5	base64 encoded MD5 value
lastModified	ISO 8601 format date and time with sub-second resolution

This is an example of a listing of etc documents:

```
[
  {
    "name": "policy.json",
    "etag": "786f68eed2afb0cd82ca325938a68ba1",
    "contentMd5": "SEkMcN3Q7wjtE2pek0tkYg==",
    "lastModified": "2014-01-08T22:15:23.725100Z"
  },
  {
    "name": "idsys.json",
    "etag": "3d5561edcecc6ea54d577fafcf0effc2",
    "contentMd5": "pYbOrt187VZtQzUsnATMQw==",
    "lastModified": "2014-01-08T22:05:44.826300Z"
  }
]
```

Token Response Formats

There are two types of token response formats: individual token GET/PUT response and token listing response. Both response formats are JSON documents. The token listing response is a JSON array of individual token objects.

owner	User name for whom this token applies
scope	Root (blank), tenant, or storage domain scope
token	Token identification; opaque value
secret	Optional S3 secret key; supplied during create
expiration	Token expiration in ISO 8601 format date and time with sub-second resolution
creation	Token creation in ISO 8601 format date and time with sub-second resolution

This is an example of a token response:

```
{
  "owner": "gcarlin",
  "scope": "",
  "token": "874fbb09057bc6be295fbdf4155deb73",
  "secret": "BaseballVsFootball",
  "expiration": "2016-02-05T04:05:55.000Z",
  "creation" : "2013-12-05T01:02:22.000Z"
}
```

Token Listing Response Format

The format of the token listing response is a JSON array of token objects. By default, the token listing response does not include the `secret` fields for the tokens.

Namespace Structure

The Content Management API namespace structure exists for every storage domain handled by the Gateway. It is a global URI mapping for all requests that come through the Gateway. The URI base for the Management API is:

URI base for Management API

`/_admin/manage/`

Below are the URI suffixes along with the HTTP methods and the corresponding Policy actions for each. If a Policy action is blank, the method is always allowed by non-authenticated requests.

Note

User-defined names supplied by the application, such as a tenant or domain name, are surrounded with curly braces, such as `{tenant}` or `{domain}`.

Management URI Methods and Policy Actions

URI Suffix	HTTP Methods	Policy Actions
Only Root Policy		
version	GET	
tenants	GET	ListTenants
meter/usage	GET	ListTenants
meter/status	GET	
tenants/{tenant}	PUT	CreateTenant
Merger of Root + Tenant Policy		
tenants/{tenant}	GET	GetTenant
	DELETE	DeleteTenant
tenants/{tenant}/meter/usage	GET	GetTenant
tenants/{tenant}/etc	GET	ListEtc
tenants/{tenant}/etc/{document}	PUT	PutPolicy
	GET	GetPolicy
	DELETE	DeletePolicy

tenants/{tenant}/tokens	GET	ListTokens
	POST	CreateToken
tenants/{tenant}/tokens/{token}	GET	ValidateToken
	DELETE	DeleteToken
tenants/{tenant}/domains	GET	ListDomains
tenants/{tenant}/domains/{domain}	PUT ⁽¹⁾	CreateDomain
Merger of Root + Tenant + Domain Policy		
tenants/{tenant}/domains/{domain}	PUT ⁽¹⁾	PutDomain
	GET	GetDomain
	DELETE	DeleteDomain
tenants/{tenant}/domains/{domain}/meter/usage	GET	GetDomain
tenants/{tenant}/domains/{domain}/etc	GET	ListEtc
tenants/{tenant}/domains/{domain}/etc/{document}	PUT	PutPolicy
	GET	GetPolicy
	DELETE	DeletePolicy
tenants/{tenant}/domains/{domain}/uuid	GET	GetDomain
tenants/{tenant}/domains/{domainUUID}/name	GET	GetDomain
Merger of Root + Tenant + Domain + Bucket Policy		
tenants/{tenant}/domains/{domain}/buckets/{bucket}/uuid	GET	GetBucket
tenants/{tenant}/domains/{domain}/buckets/{bucketUUID}/name	GET	GetBucket

Note 1: The policy action for the PUT method on the `/_admin/manage/tenants/{tenant}/domains/{domain}` URI depends upon whether or not the storage domain already exists. If the domain is being created (does not exist), CreateDomain can only be granted at the root or tenant scope and controls who can create a new domain. If the domain already exists, PutDomain controls who may change the domain and this can be granted at the root, tenant, or domain level.

Example: Getting the Management API version

```
GET /_admin/manage/version
Host: anydomain.cloud.example.com
```

The URI namespace table includes the appropriate Policy documents merged together when evaluating the access control policy for Management API requests. For example, to create a storage domain for a tenant, the **Root** and **Tenant** Policy documents are merged together. To manipulate a storage domain after it is already created, the **Root**, **Tenant**, and **Domain** Policy documents are all merged together.

System Tenant

Because the use of tenants is optional and because Swarm storage clusters may have existing storage domains created outside of Gateway, there is a concept called the **SYSTEM TENANT** containing all storage domains in the cluster not assigned to a specific tenant. These are called untenanted storage domains and, for the purpose of API consistency, these storage domains are organized within a synthetic tenant named "`_system`" in the Management API.

Unlike other tenants, the system tenant does not have an owner, an IDSYS definition, a Policy, or authentication tokens. All domains within the system tenant are subject to the inheritance rules for the root IDSYS and Policy. These untenanted domains fall under the `/_admin/manage/tenants/_system/` URI path of the Management API.

Example: Listing untenanted storage domains

```
GET /_admin/manage/tenants/_system/domains/
Host: anydomain.cloud.example.com
```

System Domain

For applications using untenanted, unnamed object content (created before the introduction of domains and named objects in Swarm), this content is accessed using the [System domain](#). For the purpose of API consistency, this content is organized within a synthetic domain named "`_system`" in the Management API.

Unlike other domains, the System domain does not have an owner or authentication tokens and does not support buckets. Domain UUID and name resolution are also not applicable. The System domain falls within the `/_admin/manage/tenants/_system/domains/_system` URI path of the Management API.

The following **Management URI Methods and Policy Actions** are not available for the System domain:

URI Suffix	HTTP Methods	Policy Actions
Merger of Root + Tenant + Domain Policy		
tenants/{tenant}/domains/ _system / uuid *Domain UUID is not applicable to System domain	GET	GetDomain
tenants/{tenant}/domains/ {domainUUID} / name *Domain Name is not applicable to System domain	GET	GetDomain
Merger of Root + Tenant + Domain + Bucket Policy		
*Bucket Policy is not applicable to System domain	GET	GetBucket

Request Methods for Buckets

This is detailed information about the request methods for buckets. Using these methods with cURL has this format (here, for listing):

```
curl -i -u caringoadmin:pwd https://site.example.com/_admin/manage/tenants/t1/domains/d1.site.exa
```

- [List Buckets](#)
- [Create Bucket](#)
- [Read Bucket](#)
- [Delete Bucket](#)
- [List Bucket ETC Documents](#)
- [Create Bucket ETC Document](#)
- [Read Bucket ETC Document](#)
- [Delete Bucket ETC Document](#)
- [Get Bucket UUID by Name](#)
- [Get Bucket Name by UUID](#)

See [Defined ETC Documents](#) for more on ETC documents (IDSYS, Policy, XFORM).

List Buckets

Method	GET
URI Suffix	tenants/{tenant}/domains/{domain}/buckets
Query Args	
Headers	
Policy Action	ListBuckets
Description	Returns a list of buckets in the domain
Restrictions	
Request Body	
Response	JSON-formatted listing response

Create Bucket

Method	PUT
URI Suffix	tenants/{tenant}/domains/{domain}/buckets/{bucket}
Query Args	
Headers	Optional metadata to be saved with bucket
Policy Action	CreateBucket

Description	Create a bucket named {bucket}. If it already exists, this action overwrites the metadata for the bucket.
Restrictions	Name must be 7-bit ASCII characters in the set [a-z, 0-9, hyphen]
Request Body	
Response	JSON general request response

Read Bucket

Method	GET
URI Suffix	tenants/{tenant}/domains/{domain}/buckets/{bucket}
Query Args	
Headers	
Policy Action	GetBucket
Description	Reads a bucket object and metadata.
Restrictions	
Request Body	
Response	Bucket object body (normally null) and metadata

Delete Bucket

Method	DELETE
URI Suffix	tenants/{tenant}/domains/{domain}/buckets/{bucket}
Query Args	recursive=yes (required)
Headers	
Policy Action	DeleteBucket
Description	Deletes all data related to the bucket
Restrictions	
Request Body	
Response	JSON general request response

List Bucket ETC Documents

Method	GET
URI Suffix	tenants/{tenant}/domains/{domain}/buckets/{bucket}/etc
Query Args	
Headers	
Policy Action	ListEtc
Description	Returns a list of bucket documents
Restrictions	Paging is not supported in the request; only 1000 returned
Request Body	
Response	JSON-formatted documents listing response

Create Bucket ETC Document

Method	PUT
URI Suffix	tenants/{tenant}/domains/{domain}/buckets/{bucket}/etc/{document}
Query Args	
Headers	Any metadata to be included with the document
Policy Action	PutPolicy
Description	Create or overwrite a document associated with the bucket
Restrictions	Maximum document size is 1MB
Request Body	The document contents
Response	JSON general request response

Read Bucket ETC Document

Method	GET
URI Suffix	tenants/{tenant}/domains/{domain}/buckets/{bucket}/etc/{document}
Query Args	
Headers	

Policy Action	GetPolicy
Description	Read a bucket document
Restrictions	
Request Body	
Response	Document body and metadata

Delete Bucket ETC Document

Method	DELETE
URI Suffix	tenants/{tenant}/domains/{domain}/buckets/{bucket}/etc/{document}
Query Args	
Headers	
Policy Action	DeletePolicy
Description	Delete a bucket document
Restrictions	
Request Body	
Response	JSON general request response

Get Bucket UUID by Name

Method	GET
URI Suffix	tenants/{tenant}/domains/{domain}/buckets/{bucket}/uuid
Query Args	
Headers	
Policy Action	GetBucket
Description	Returns the UUID (context ID) for a bucket
Restrictions	
Request Body	
Response	JSON general request response

Get Bucket Name by UUID

Method	GET
URI Suffix	tenants/{tenant}/domains/{domain}/buckets/{bucketUUID}/name
Query Args	
Headers	
Policy Action	GetBucket
Description	Returns the canonical name for a bucket
Restrictions	
Request Body	
Response	JSON general request response

Defined ETC Documents

The Content Gateway makes use of the **etc** document storage for tenants and storage domains to store IDSYS, Policy, and XFORM information. These defined document names are used by the Gateway and are exposed through the Management API as an end-point for integration with applications.

- [IDSYS](#)
- [Policy](#)
- [XFORM](#)

IDSYS

The IDSYS documents for tenants and storage domains are created and modified by uploading the JSON document through the Management API.

```
tenants/{tenant}/etc/idsys.json  
tenants/{tenant}/domains/{domain}/etc/idsys.json
```

The entire JSON document with all fields must be provided when updating the **idsys.json** document and the **Content-Type: application/json** header must be included with the request.

- Permission to create and update is granted with the **PutPolicy** policy action.
- Reading the IDSYS document is controlled with the **GetPolicy** policy action.

The storage domain's IDSYS can also be manipulated through the SCSP Storage API.

Policy

The Policy documents for tenants and storage domains are created and modified by uploading the JSON document through the Management API.

```
tenants/{tenant}/etc/policy.json  
tenants/{tenant}/domains/{domain}/etc/policy.json
```

The entire JSON document with all fields must be provided when updating the `policy.json` document and the **Content-Type: application/json** header must be included with the request.

- Permission to create and update is granted with the **PutPolicy** policy action.
- Reading the Policy document is controlled with the **GetPolicy** policy action.

The access control policies for domains and buckets can also be manipulated through the SCSP Storage API.

XFORM

The metadata transform (XFORM) document for storage domains is created and modified by uploading the JSON document through the Management API.

```
tenants/{tenant}/domains/{domain}/etc/metaxform.json
```

The entire JSON document with all fields must be provided when updating the `metaxform.json` document and the **Content-Type: application/json** header must be included with the request.

- Permission to create and update is granted with the **PutPolicy** policy action.
- Reading the XFORM document is controlled with the **GetPolicy** policy action.

The metadata transform can also be manipulated through the SCSP Storage API.

Methods for Quotas

Set and clear quotas and check on quota statuses using the Content Management API in addition to specifying quota policies directly in the Content Portal:

See [Setting Quotas](#) and [Content Metering](#).

Legend:

- {M} = metric name, one of "bandwidth", "rawstorage", "storage"
- {T} = tenant name
- {D} = domain name
- {B} = bucket name

Method and Suffix	Query Arguments	Policy Action	Notes
PUT /_admin/manage/tenants/{T}/ <ul style="list-style-type: none"> • quota/{M}/limit • domains/{D}/quota/{M}/limit • domains/{D}/buckets/{B}/quota/{M}/limit 	limit={integer}{KB MB GB TB} state={ok notify nowrite read lock} None removes current values	PutQuota	Sets or clears a limit and state. If this results in a state change, then email notifications are sent. *Not applicable to the System domain Affected headers: <ul style="list-style-type: none"> • x-caringo-meta-quota-{M}-limit • x-caringo-meta-quota-{M}-current
PUT /_admin/manage/tenants/{T}/ <ul style="list-style-type: none"> • quota/{M}/override • domains/{D}/quota/{M}/override • domains/{D}/buckets/{B}/quota/{M}/override 	duration={number}{s m d w} deadline={timestamp} statduration={ok notify nowrite read lock} None removes current values	PutQuota	Sets or clears an override. If this results in a state change, then email notifications are sent. Duration is a number plus a unit suffix for seconds, minutes, days, or weeks. Deadline is a timestamp in ISO 8601 format, such as 2016-07-01T00:00:00Z, which specifies when the override expires. *Not applicable to the System domain Affected headers: <ul style="list-style-type: none"> • x-caringo-meta-quota-{M}-override
PUT /_admin/manage/tenants/{T}/ <ul style="list-style-type: none"> • quota/email • domains/{D}/quota/email • domains/{D}/buckets/{B}/quota/email 	addresses={email}[email],...] None removes current values	PutQuota	Sets or clears one or more email recipients for quota notifications. Updates the header on the context object, where multiple email addresses are collapsed into a single comma-separated list. The list is always replaced as a whole. *Not applicable to the System domain Affected headers: <ul style="list-style-type: none"> • x-caringo-meta-quota-email

<p>HEAD /_admin/manage/tenants/{T}/</p> <ul style="list-style-type: none"> • quota/{M} • domains/{D}/quota/{M} • domains/{D}/buckets/{B}/quota/{M} • quota/* • domains/{D}/quota/* • domains/{D}/buckets/{B}/quota/* 	-	GetQuota	<p>Retrieves quota information for a single metric or all (*) metrics of a specific context (tenant, domain, bucket).</p> <p>If a metric {M} has no limit or override configured, then the corresponding current/limit/override headers are not present in the response. If the context has no quota config for any metric, no quota headers are returned.</p> <p>*Not applicable to the System domain</p> <p>Affected headers:</p> <ul style="list-style-type: none"> • x-caringo-meta-quota-{M}-current = {computed state}; {actual usage}; {timestamp} • x-caringo-meta-quota-{M}-limit = {state}; {limit} • x-caringo-meta-quota-{M}-override = {state}; {user}; {deadline} • x-caringo-meta-quota-{M}-refreshdelay = {milliseconds} • x-caringo-meta-quota-email = {comma-separated-list}
<p>HEAD /_admin/manage/tenants/{T}/</p> <ul style="list-style-type: none"> • quota/check • domains/{D}/quota/check • domains/{D}/buckets/{B}/quota/check 	-	GetQuota	<p>Performs a quota status check, which takes into account parent/child context relationships and overrides.</p> <p>*Not applicable to the System domain</p> <p>Affected headers:</p> <ul style="list-style-type: none"> • x-caringo-meta-quota-metric • x-caringo-meta-quota-context-type • x-caringo-meta-quota-context-name • x-caringo-meta-quota-type • x-caringo-meta-quota-state • x-caringo-meta-quota-message
<p>GET /_admin/manage/quota/status</p>	-	ListDomain	<p>Performs a check whether the quota feature is enabled.</p> <p>*Not applicable to the System domain</p> <p>Returns JSON body {"enabled":true}</p>
<p>GET /_admin/manage/tenants/{T}/</p> <ul style="list-style-type: none"> • domains/{D}/meter/usage/bytesIn • domains/{D}/meter/usage/bytesOut 	<p>from={timestamp}</p> <p>to={timestamp}</p>		<p>Retrieves the current bandwidth usage between a date range. The bandwidth metric is a special case; it is reset at the beginning of each month. The corresponding Elasticsearch queries must adapt the from/to interval to take this into account.</p> <p>Each argument is a timestamp in ISO 8601 format, such as 2016-07-01T00:00:00Z.</p>

<p>GET /_admin/manage/tenants/{T}/</p> <ul style="list-style-type: none"> • meter/usage/bytesSize/current • domains/{D}/meter/usage/bytesSize/current • domains/{D}/buckets/{B}/meter/usage/bytesSize/current • meter/usage/bytesStored/current • domains/{D}/meter/usage/bytesStored/current • domains/{D}/buckets/{B}/meter/usage/bytesStored/current 	-		Retrieves the current logical or raw storage through a point-in-time query.
---	---	--	---

Domain Adoption

It is possible to have a tenant adopt a storage domain so it can be accessed through the Content Portal if the storage domain is created outside the Content Management API (such as a replication cluster). There is a special variation to the domain adoption procedure, noted below if that storage domain was created with Swarm's legacy auth/auth.

Verify the following for domain adoption:

- The tenant exists.
- The cluster administrator executes these actions directly against the storage cluster and *not* through the Gateway.



Note

This is a highly privileged operation that has no equivalent request within the Gateway.

To adopt a domain

1. Retrieve all custom metadata attached to the domain.
2. Update the domain providing all custom metadata and an `x-tenant-meta-name` header.

This is an example of the commands. The strings {tenant} and {domain} are substituted for the actual tenant name and storage domain name.

The first step is to retrieve all current, custom metadata name/value pairs for the domain:

```
HEAD /?domain={domain}
```

There is a `Castor-Authorization` header in the response or a "401 Unauthorized" response to the previous request may be received if the domain has Swarm's legacy auth/auth on it. See the section below for instructions to remove the legacy auth/auth.



Deprecated

The native Swarm auth/auth feature is deprecated and is removed as of June 2017.

Certain field names are valid as custom metadata. In general, use `Castor-*` (except for `Castor-System-*`), `Content-*`, `X-*-Meta`, and `X-*-Meta-*` headers in the **HEAD** response as custom metadata for the storage domain. With the exception of `Castor-Authorization`, these are the fields to preserve.

For details about headers, see [SCSP Headers](#) and [SCSP COPY](#).

Use the **COPY** request to replace all object metadata in the storage domain and include the adoptive tenant's name after retrieving all custom metadata name/value pairs (denoted as {mdName#} and {mdValue#}) from the **HEAD** request:

```
COPY /?domain={domain}&replicate=immediate
  x-tenant-meta-name: {tenant}
  {mdName1}: {mdValue1}
  {mdName2}: {mdValue2}
  ...
```

The `x-tenant-meta-name` *must* match the name of an existing tenant created through the Gateway Management API or Content Portal.

Upon completion of the domain adoption procedure, the storage domain is now subject to the tenant access control policy in addition to the root and domain policies. The storage domain switches to using the tenant IDSYS rather than the root IDSYS if the storage domain does not define a separate IDSYS, was previously using the root IDSYS, and the adoptive tenant defines an IDSYS.

Removing legacy auth/auth

Legacy auth/auth needs to be removed so the domain can be used correctly through Gateway if the storage domain has legacy auth/auth on it. These examples use the curl command line utility since it is able to perform HTTP digest authentication. These examples can be adapted for use with another tool or library to issue the **HEAD** and **COPY** commands. Since this process is very similar to the previous one for domains without legacy auth/auth, references are made to the instructions from the previous section. The {adminUser} username in these examples must be for one of the Swarm administrators defined in the storage cluster's configuration. The {storageNode} string is the host or IP for any node in the storage cluster.

Get the current metadata for the storage domain.

```
curl -I --digest -u {adminUser}
--location-trusted
'http://{storageNode}/?domain={domain}'
```

All custom metadata name/value pairs are needed with the exception of the `Castor-Authorization` header.

The previously described **COPY** request is performed using HTTP digest authentication.

```
curl -X COPY --digest -u {adminUser}
--location-trusted
-H 'x-tenant-meta-name: {tenant}'
-H '{mdName1}: {mdValue1}'
-H '{mdName2}: {mdValue2}'
...
'http://{storageNode}/?domain={domain}&replicate=immediate'
```

All prior discussion about the `x-tenant-meta-name` value and post-creation domain behavior apply.

Content SCSP Extensions

This section documents the Content Gateway enhancements to the Swarm SCSP client protocol.

**Note**

These SCSP protocol changes are only applicable when communicating to the object storage cluster through the Gateway.

- [SCSP Context Sub-resources](#)
- [Gateway ACL for Objects](#)
- [Recursive Deletes](#)
- [Multipart MIME POST](#)
- [Gateway CORS for Buckets](#)
- [Domain and Bucket Creation](#)

SCSP Context Sub-resources

The Gateway creates SCSP context sub-resources to allow the specification of identity management systems, access control policies, and metadata transforms.

These are the sub-resources and the context in which they are applicable when using the Gateway.

Sub-resource	Context	Description
idsys	domain	Identity system definition
policy	domain, bucket	Access control policy
xform	domain, bucket	Metadata transform

All storage domain and bucket sub-resources are controlled with one of the policy actions **PutPolicy**, **GetPolicy**, or **DeletePolicy**.



Warning

Permission to read or change these sub-resources for a storage domain must be protected from untrusted users and, in deployments where end-users are allowed to manage storage domains, a cluster or tenant administrator normally retains ownership of the storage domain. An end-user is able to read and change the domain's sub-resources if they own the storage domain.

- [Warning](#)
- [IDSYS](#)
- [Policy](#)
- [XFORM](#)

IDSYS

The IDSYS document sub-resource for a storage domain is manipulated using authenticated SCSP commands through the Gateway. This is accomplished by uploading the JSON document for the IDSYS to the storage domain's **idsys** sub-resource using the HTTP PUT operation.

```
PUT /?idsys Content-Type: application/json
{"ldap" : {
  "ldaphost" : "ldap.example.com", ...
}}
```

The entire JSON document with all fields must be provided when updating the **idsys** sub-resource and the **Content-Type: application/json** header must be included with the request.

Permission to update the IDSYS document for a domain is granted with the **PutPolicy** policy action.

Reading the IDSYS document is controlled with the **GetPolicy** policy action and uses the **HTTP GET** operation.

```
GET /?idsys
```

An IDSYS is removed using the HTTP DELETE operation and controlled with the **DeletePolicy** policy action.

```
DELETE /?idsys
```

Policy

The Policy document sub-resources for storage domains and buckets are manipulated using authenticated SCSP commands through the Gateway.

Creating a new Policy document or replacing an existing one are both controlled with the **PutPolicy** action. The entire JSON document with all fields must be provided when updating the **policy** sub-resource and the **Content-Type: application/json** header must be included with the request.

The HTTP PUT operation is used to update a domain Policy:

```
PUT /?policy
Content-Type: application/json
{"Id": "My Domain Policy", ... }
```

...or a bucket Policy:

```
PUT /mybucket?policy
Content-Type: application/json
{"Id": "My Bucket Policy", ... }
```

Reading a Policy document is controlled by the **GetPolicy** action. Examples of reading a Policy for a storage domain and a bucket:

```
GET /?policy
GET /mybucket?policy
```

Deleting a Policy document is controlled by the **DeletePolicy** action. Examples of deleting a Policy for a storage domain and a bucket:

```
DELETE /?policy
DELETE /mybucket?policy
```

XFORM

The metadata transform (XFORM) sub-resource for domains and buckets are manipulated using authenticated SCSP commands through the Gateway.

Creating a new XFORM document or replacing an existing one are both controlled with the **PutPolicy** action. The entire JSON document with all fields must be provided when updating the **xform** sub-resource and the **Content-Type: application/json** header must be included with the request.

The HTTP PUT operation is used to update a domain XFORM:

```
PUT /?xform
{"metadata" : { ... }}
```

Or a bucket XFORM:

```
PUT /mybucket?xform
{"metadata" : { ... }}
```

Reading an XFORM document is controlled by the **GetPolicy** action. Examples of reading an XFORM for a storage domain and a bucket:

```
GET /?xform
GET /mybucket?xform
```

Deleting an XFORM document is controlled by the **DeletePolicy** action. Examples of deleting an XFORM for a storage domain and a bucket:

```
DELETE /?xform
DELETE /mybucket?xform
```

Gateway ACL for Objects

- [Differences from S3](#)
- [GET Object ACL](#)
- [GET Object Version ACL](#)
- [PUT Object ACL](#)
- [PUT Object Version ACL](#)

Gateway supports management of access control lists (ACLs) for objects, including changes to existing authorizations.



Differences from S3

- `PUT /object?acl&versionId=X` cannot be used because metadata on old versions is immutable.
- Updating an object's acl also updates the object's modification time.
- Gateway uses a convention whereby user names are decorated with `{username}@{domain}` or `{username}+{tenant}` depending on the idsys in which the user is defined. If the user is defined in the root idsys, then the decoration looks like `{username}@`. This decoration may be omitted if there cannot be any ambiguity, but, internally, Gateway always stores decorated usernames in the ACL owner and user grantees, adding it to an incoming ACL as needed and removing it where possible before passing an ACL back to the client.

GET Object ACL

GET Object acl uses the **acl** subresource to return the access control list (ACL) of an object. To use this operation with S3, you must have `READ_ACP` access to the object.

The following request returns information, including the ACL, of an object:

```
GET /{object-name}?acl HTTP/1.1
```

GET Object Version ACL

The following request returns information, including the ACL, of a specific version of the object:

```
GET /{object-name}?version={etag}&acl HTTP/1.1
```

PUT Object ACL

PUT Object acl uses the **acl** subresource to send the ACL of an object in the request body (rather than in the request headers):

```
PUT /{object-name}?acl HTTP/1.1
```

```
<AccessControlPolicy>
  <Owner>
    <ID>ID</ID>
    <DisplayName>EmailAddress</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
        <ID>ID</ID>
        <DisplayName>EmailAddress</DisplayName>
      </Grantee>
      <Permission>Permission</Permission>
    </Grant>
    . . .
  </AccessControlList>
</AccessControlPolicy>
```

PUT Object Version ACL

The following request returns information, including the ACL, of a specific version of the object:

```
PUT /{object-name}?version={etag}&acl HTTP/1.1
```

Recursive Deletes

Gateway utilizes Swarm's recursive deleting mechanism with some augmentation for requests not containing a "recursive={value}" query argument.

See [Managing Domains](#) and [SCSP Query Arguments](#) for more about recursive deletes of domains and buckets.

Gateway automatically adds "recursive=yes" to the DELETE request transmitted to Swarm when removing an empty storage domain or an empty bucket if the client request does not include the recursive query argument. This preserves the protocol behavior of the Gateway versions prior to 4.0. This is analogous to the Unix `rmdir` behavior on empty directories.

The request must include the recursive query argument or else the request fails if a client issues a DELETE on a non-empty storage domain or bucket. This is analogous to attempting to run `rmdir` on a non-empty directory in Unix.

The Gateway client audit log records the single DELETE operation. The individual objects do not receive individual audit log records of deletion even if the delete is permitted and the domain or bucket contains objects.

Delay

Some deletes may be subject to a delay period although Swarm's recursive deletes are always asynchronous. See [SCSP Query Arguments](#) for explanation of the "recursive=yes" and "recursive=now" delete options.

Important

Content UI uses the "recursive=now" delete option for storage domain and bucket removal.

Multipart MIME POST

Content Gateway allows client applications to use the HTTP multipart MIME POST to upload multiple files in one operation. Gateway converts these multiple parts into individual POST operations to Swarm. While Gateway always returns an HTTP 202 response code, the body of the response contains the results of the individual POST operations. The Content Portal uses this mechanism for the upload page.

While processing this type of request, the individual files are extracted from the original POST request and spooled to the Gateway server's local file system before transmitting them to Swarm. The spool directory is specified with the `multipartSpoolDir` setting and is allowed to fill the file system up to a maximum percentage defined with the `multipartUsageAllowed` setting.

 **Important**

Verify the file system has sufficient disk space available to handle the incoming requests.

The multipart MIME POST request is the only type of request that uses a local disk spool on the Gateway. Requests such as SCSP single-object writes, S3 multipart uploads, and [SCSP multipart writes](#) are all streamed directly to Swarm.

Gateway CORS for Buckets

- [Note](#)
- [Enabling CORS on a Bucket](#)
- [CORSRule Elements](#)

Gateway supports Cross-Origin Resource Sharing (CORS) so a specific bucket can be accessed by a web page in a different domains. Configure a bucket to allow cross-origin resource access by using CORS configuration rules. These are two common scenarios for using CORS:

- **Outbound Access** - Hosting a website in a bucket, but the site pages to use style sheets, images, and scripts are managed elsewhere. Because browsers block such requests from within scripts, configure the bucket to explicitly enable cross-origin requests.
- **Inbound Access** - Hosting a public resource from a bucket. Because browsers require a CORS check (known as a *preflight check*), configure the bucket to allow *any* origin to make these requests.

See the [W3C specification for CORS](#).



Note

Set CORS configuration using S3, not SCSP. A browser accessing the bucket receives the same CORS information in the response from both S3 and SCSP.

Enabling CORS on a Bucket

To configure a bucket to allow cross-origin requests, create a CORS configuration, an XML document with up to 100 rules identifying the origins that can access a bucket, the operations (HTTP methods) to support for each origin, and other operation-specific information. Add the XML document as the **cors** subresource to the bucket.

This **cors** configuration on a bucket has three rules (the **CORSRule** elements), which do the following:

1. Allow cross-origin PUT, POST, and DELETE requests from the `https://www.example1.com` origin and allow all headers in a preflight OPTIONS request through the Access-Control-Request-Headers header. In response to any preflight OPTIONS request, Gateway returns any requested headers.
2. Allow the same cross-origin requests as the first rule but to another origin, `https://www.example2.com`.
3. Allow cross-origin GET requests from all origins. The `*` wildcard character refers to all origins.

```

<CORSConfiguration>
  <CORSRule>
    <AllowedOrigin>http://www.example1.com</AllowedOrigin>
    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>
    <AllowedHeader>*</AllowedHeader>
  </CORSRule>
  <CORSRule>
    <AllowedOrigin>http://www.example2.com</AllowedOrigin>
    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>
    <AllowedHeader>*</AllowedHeader>
  </CORSRule>
  <CORSRule>
    <AllowedOrigin>*</AllowedOrigin>
    <AllowedMethod>GET</AllowedMethod>
  </CORSRule>
</CORSConfiguration>
    
```

The CORS configuration allows optional configuration parameters, as shown in this CORS configuration that allows cross-origin PUT and POST requests from `http://www.example.com`:

```

<CORSConfiguration>
  <CORSRule>
    <AllowedOrigin>http://www.example.com</AllowedOrigin>
    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>
    <AllowedHeader>*</AllowedHeader>
    <MaxAgeSeconds>3000</MaxAgeSeconds>
    <ExposeHeader>x-amz-server-side-encryption</ExposeHeader>
    <ExposeHeader>x-amz-request-id</ExposeHeader>
    <ExposeHeader>x-amz-id-2</ExposeHeader>
  </CORSRule>
</CORSConfiguration>
    
```

CORSRule Elements

Element	Description
AllowedMethod	Specifies which of the following values is allowed: GET, PUT, POST, DELETE, HEAD
AllowedOrigin	Specifies the origins cross-domain requests are allowed from: <code>http://www.example.com/</code> . The origin string can contain at most one * wildcard character, such as <code>http://*.example.com</code> . Optionally specify * as the origin to enable all origins to send cross-origin requests. Specify <code>https</code> to enable only secure origins.
AllowedHeader	Specifies which headers are allowed in a preflight request through the Access-Control-Request-Headers header. Each header name in the Access-Control-Request-Headers header must match a corresponding entry in the rule. Gateway sends only the allowed headers in a response requested. Each AllowedHeader string in the rule can contain at most one * wildcard character. <code><AllowedHeader>x-amz-*</AllowedHeader></code> enables all Amazon-specific headers.
ExposeHeader	Identifies a header in the response clients are able to access from applications (e.g. from a JavaScript XMLHttpRequest object).

MaxAgeSeconds

Specifies the time in seconds a browser can cache the response for a preflight request as identified by the resource, the HTTP method, and the origin. By caching the response, the browser does not need to send preflight requests if the original request is to be repeated.

Domain and Bucket Creation

Swarm Storage has two types of context objects: domains and buckets. When you create them from Gateway, follow these guidelines:

Required Header

Gateway requires that the HTTP POST request include the following header to identify the content type as a context object when clients create these contexts:

`Content-Type: application/castorcontext`

The Gateway returns an HTTP 400 Bad Request error response to the client if contexts are not explicitly identified as shown during creation.



Note

Contexts that already exist in the storage cluster or that are created directly to the cluster do not require this **Content-Type** specification to work properly with Gateway.

Required Argument

SCSP requires you to add the [domain query argument](#) when creating a storage domain, regardless of the existence of a `X-Forwarded-Host` or `Host` header on the request.

Optional Sub-resources

Gateway supports the common application need to immediately create one or more of the IDSYS, Policy, and XFORM sub-resources for the new context in a one-shot request. This is performed by including the additional documents within a JSON formatted request body on the context create request. If you do not need this one-shot creation of authentication for the context, the context is created with an HTTP POST request that is submitted with a null request body, `Content-Length` zero, and `Content-Type` of `application/castorcontext`.

The format of the JSON body of a one-shot creation request must conform to the following JSON schema.

For more about JSON schemas, see json-schema.org.

```

{
  "type": "object",
  "properties": {
    "config": {
      "type": "object",
      "properties": {
        "idsys": {
          "$ref": "https://support.cloud.caringo.com/schemas/idsys-schema.json#"
        },
        "policy": {
          "$ref": "https://support.cloud.caringo.com/schemas/policy-schema.json#"
        },
        "metaxform": {
          "$ref": "https://support.cloud.caringo.com/schemas/metaxform-schema.json#"
        }
      },
      "optional": true,
      "additionalProperties": false
    },
    "metadata": {
      "type": "object",
      "optional": true
    }
  },
  "additionalProperties": false
}
    
```

This is an example JSON document using the schema:

```

{
  "config": {
    "idsys": { ... },
    "policy": { ... },
    "metaxform": { ... }
  },
  "metadata": {
    "X-Copyright-Meta": "Copyright 2015 Widgets, Inc."
  }
}
    
```

Any combination of the IDSYS, Policy, and XFORM document sub-resources can be included in the one-shot request. IDSYS is only valid for a storage domain context and not for bucket contexts. The same configuration document formats described in this guide are used within the "{ ... }" portions of the example.

Updates to its IDSYS, Policy, or XFORM sub-resources must be done individually and cannot use the composite form of the one-shot creation request after a context object has been created.

Swarm SDK

- [SDK for C++](#)
- [SDK Overview](#)
- [SDK for C#](#)
- [SDK for Python](#)
- [SDK for Java](#)

SDK for C++

- [C++ SDK Installation and Packaging](#)
- [Building the C++ Client on Linux](#)
- [Building the C++ Client on Windows](#)
- [Implementation Notes](#)
- [Using the C++ Client Sample Code](#)
- [C++ Administrative Override of an Allow Header](#)

C++ SDK Installation and Packaging

The C++ distribution contains the following directory structure:

- `docs`: Contains HTML documentation for the C++ implementation
- `examples`: Contains example source and makefile which compiles and runs against the supplied libraries and headers.
- `src`: Contains source and makefile
 - `requestHandler`: single-request HTTP communication engine
- `ScspCPPWin`: Contains the Visual Studio project for the Windows C++ implementation

Building the C++ Client on Linux

The C++ client requires these libraries and utilities:

- `libcurl`
- GNU Compiler Connection (`gcc`)

libcurl Requirements for the C++ Client

The C++ client requires a particular version of `libcurl` for the operating system.

The Swarm SDK is built on `libcurl`, version 7.20.1 or later. Version 7.21.2 was used for primary testing. `libcurl` is available pre-packaged or in source from <https://curl.se/libcurl/>. Obtain version 7.20.1 or later source archives listed at the top of the [curl download page](#).

Not all platforms ship with a recent version of `libcurl` by default:

- SUSE: The packaged `curl` for SUSE 11.2 is version 7.20.1.6, which is supported by the SDK; the additional `libcurl-devel` package must be installed.
- Windows 2003 and 2008: Get a recent version [here](#).

GNU Compiler Connection (`gcc`) Requirement for the C++ Client

- [gcc version 4 or later](#) is required for Linux. Version 4.4.0 was used in primary testing.

Compiling the C++ Client on Linux

Use shell scripts that call `make` files to build the source for a Linux target platform.

Run the following from the `src` directory to build the source for Red Hat Enterprise Linux (RHEL) or CentOS:

- ./makeAllRH from the src directory

Run the following from the src directory to build the source for SUSE:

- ./makeAllSU

Building the C++ Client on Windows

DataCore tested the SDK with Microsoft Visual Studio Express 2008. Additional tasks may be required if using a different version of Visual Studio. Consult the documentation provided with Visual Studio for more information.

Compiling cURL on Visual Studio Express 2008

To compile cURL on Visual Studio Express 2008:

1. Download the most recent version of curl [available on the curl download page](#).
2. Open the curl Visual Studio project (it may be named `vc6curl.dsw`).
3. The project must be converted to Visual Studio 2008 format.
4. Build curl.
Because the procedure to compile curl changes frequently, consult an online reference such as the [libcurl install page](#) for more information.
5. After the solution builds successfully, close the project.

Compiling the Swarm SDK Using Visual Studio Express 2008

To compile the SDK using Visual Studio Express 2008:

1. Extract the SDK .zip file into an empty folder.
2. Open the Visual Studio solution (`ScspCPPWin.sln`).
3. In the Solution Explorer pane, right-click the **ScspCPPWin** solution.
4. From the pop-up menu, click **Properties**.
5. From the Configuration list, click **Debug**.
6. Expand **Configuration Properties > C/C++ > General**.
7. In the **Additional/Include Directories** field, browse to locate the libcurl include directory.
8. Apply the changes and build the solution.
9. Repeat these tasks for the release project.

Compiling Custom Applications Using Visual Studio Express 2008

This section provides basic guidelines for compiling custom applications with Visual Studio Express 2008. DataCore does not recommend any particular coding style or method; the following tasks must be included in the workflow for the compilation to succeed.

Perform the following tasks in debug and release projects to compile custom applications using Visual Studio Express 2008:

1. Right-click the solution in the Solution Explorer pane.
2. From the pop-up menu, click **Properties**.
3. Expand **Configuration Properties > C/C++ > General > Additional Includes**.
4. Include the following:

- Path to the curl include folder.
 - Path to the ScspCPP\src folder.
5. Expand **Linker > General > Additional Library Directories**.
 6. Include the following:
 - Path to the curl lib folder.
 - Path to the ScspCPP\ScspCPPWin\Debug or ScspCPP\ScspCPPWin\Release folder.
 7. Expand **Linker > Input > Additional Dependencies**.
 8. Include paths to the following:
 - ws2_32.lib wldap32.lib ScspCPPWin.lib
 - libcurl.lib (debug) or libcurl.lib (release)

Implementation Notes

Using Connection Pooling with the C++ Client

The connection pool stores open, previously used connections for reuse so clients do not need to negotiate opening a socket for every request. Configure connection pooling for the C++ client for using the `maxStoredConnections` parameter in the `ScspClient` class. DataCore recommends setting the value of this parameter to the number of threads multiplied by the number of Swarm cluster nodes.

For more information about this parameter, see the **scsp::ScspClient Class Reference** topic in the HTML documentation provided with the SDK, which is located in the `cpp/docs/html` directory in the SDK package.



Note

Limit the value to 5 times the number of threads to avoid reaching the client's operating system limits on open file descriptors for installations with a large number of nodes and a high thread count (approximately 100 or more).

Notes About the C++ SDK

Following are notes to when writing C++ code for the SDK:

- **ScspClient chunkSize parameters support in C++.** `ScspClient` in all languages, including C++, supports the following parameters: `getChunkSize`, `setChunkSize`. `cURL` does not support explicitly setting how many bytes are sent at a time. `cURL` provides a buffer and the buffer's length but does not enable the SDK to set the size of the buffer.
- **Compilation Warnings for `scspCredential.ccp`.** When compiling C++ on Visual Studio, unsafe warnings related to conversion of the stream size argument for `scspCredentials` may print. These errors can be safely ignored.
- **Expect headers.** Because of a limitation of `libcurl`, to include more than one Expect header in a header entry, include them as a comma-separated list such as the following:

```
wHeaders.addValue("Expect", "Content-MD5,100-Continue");
```

Do *not* send each Expect header as a separate request because a large number of SCSP WARNING: Please use Expect: 100-continue... errors can result.

- **Character encoding.**
 - A backslash character must be escaped with `%5c (\)` if passing in a URI path.
 - The string class must be used when passing a path as a string. If the path needs includes non-ASCII characters, these characters must be UTF-8 encoded by the caller.

- **Synchronization Classes.** To keep the SDK independent of a particular threading and synchronization package, the synchronization primitives used internally are surfaced. SDK clients must derive lock and semaphore objects from the ScspLock and ScspSemaphore and the respective factories and pass instances of these into the ScspClient constructor. There is an example null lock and semaphore implemented in `examples/ ScspExamples.cpp`. Also, there is a ScspBasicLock in `src/locator.hpp` that implements a ScspLock derivative based on a user-implemented ScspSemaphore derivative for ease of programming. See the documentation for `locator.hpp` or the Synchronization module in HTML help for more information.
- **Pointer Ownership.** Header and HTML documentation notes cover ownership of a pointer to an object. For the SDK, ownership of a pointer implies memory management responsibility. If an object owns a pointer to another object, it is responsible for deleting the object. The pointers to the ScspLockFactory and ScspSemaphoreFactory passed into the constructor of an ScspClient are owned by the client and are deleted when the ScspClient instance is deleted.
- **ScspResponse.** Most of the ScspClient commands take a pointer to a ScspResponse as an argument for performance optimization. This response object is used internally by the SDK to set query result information. See the ScspClient documentation for more information.
- **Exceptions.** By design, C++ does not throw exceptions to indicate error conditions. Any exceptions thrown are due to system signals converted to exceptions by the gcc runtime.
- **Mimetype Discovery.** curl does not support mimetype discovery, so the C++ client also does not.
- **Scsp Streaming.** The SDK defines two interfaces, `scsp_istream` and `scsp_ostream`, that encapsulate all required behavior from an input stream, from which content is retrieved for storage into a Swarm stream, or an output stream, into which to retrieve the content of a Swarm stream. Implementations are provided that wrap a `std::istream` and `std::ostream`. `scsp_istream` and `scsp_ostream` are less complex to implement than a custom Standard C++ `<iostream>` `streambuf` and can support large (>2GB) streams, even on 32-bit systems.
- **Libcurl Versions.** Versions of libcurl earlier than 7.20.1 do not fully support all components necessary to the SDK and are not certified for use.
- **Request Timeouts.** In addition to the `connectionTimeout`, the C++ implementation uses a `requestTimeout` for the time limit, in secs/GB, for the request to complete. A 2GB write request times out after 400 seconds with a `requestTimeout` of 200 secs/GB.

Using the C++ Client Sample Code

The C++ SDK client ships with runnable sample code that demonstrates how to write client applications for Swarm. The C++ sample code is located in the `sdk-zip-extract-dir/caringo-sdk-version-brand/cpp/examples` directory and consists of the following files:

C++ sample code file name	Description
ObjectEnumerator- Example.cpp	ObjectEnumerator class examples for use with the Content Router. Content Router is deprecated and these examples are planned to be removed in a future release.
ScspExample.cpp	General SCSP client examples.
ScspRemoteExample.cpp	Examples of performing SCSP operations on local and remote clusters using the SCSP Proxy.

Exploring the C++ Sample Code

The `sdk-extract-dir/SwarmSDK-version/docs/example` directory contains commented C++ sample code viewable in a web browser. Double-click `index-all.html` to open the index page in the default web browser. The commented samples include the following:

- Authenticating named objects
- Performing SCSP operations on unnamed mutable and immutable objects
- Using lifepoints and MD5 integrity seals
- Using READ, including validation, ranges, and Content-MD5

Performing SCSP operations on objects in local and remote clusters using the SCSP Proxy

The commented sample code is part of the Swarm bundle.

C++ Administrative Override of an Allow Header

This example is not included with the code samples provided with the SDK.

```
// This example shows how to execute a copy command with administrative
// override to clean up Allow headers. Note that you may want to get the
// original headers first to make sure none of the existing ones on the stream
// are lost.

string DEFAULTADMINREALM = "CASTor administrator";
isw.seekg(0, ios::beg);
ScspHeaders headers;
// set the admin credentials
ScspAuthentication auth("admin", "ourpwdofchoicehere", DEFAULTADMINREALM);
headers.setAuthentication(auth);
ScspQueryArgs args;
// make the request administrative
args.setValue("admin", "yes");
client.copy(uuid, &response, &args, &headers);
```

SDK Overview

- [Getting Started](#)
 - [Installation](#)
 - [Run-Time Configuration](#)
 - [Logging](#)
 - [Errors and Status Codes](#)
 - [Common Terminology](#)
 - [Using SDK Code Examples](#)
- [ScspClient](#)
 - [SCSP Classes](#)
 - [Support Classes](#)
 - [Validation Mode](#)
- [Managing Domains and Buckets](#)
- [Object Headers and Query Arguments](#)
 - [Using ScspHeaders for System and Custom Metadata](#)
 - [ScspQueryArgs](#)



Note

The Swarm Software Development Kit (SDK) is deprecated and replaced with an HTTP or S3 library connecting to a content gateway, therefore, it is not recommended using SDKs.

The Swarm Software Development Kit (SDK) simplifies integration with Swarm by providing client library support for handling of specific SCSP behaviors to programmers developing Swarm applications. A client application communicates with Swarm using a subset of the HTTP/1.1 protocol called Simple Content Storage Protocol (SCSP). SCSP implements all required components and most of the common methods of the HTTP protocol and includes the Swarm specific handling of standard conventions like URL query arguments and custom request headers.

The SDK describes a consistent set of features using a common API in each of the following programming languages:

- C++
- C#
- Java
- Python

All example clients are synchronous and thread-safe. High performance applications can call clients from multiple threads and/or multiple processes without interference or deadlocks.



Note

Review the [Storage SCSP Development](#) prior to using the Software Development Kit for a full understanding of the SCSP protocol and recommendations for client integration. This guide assumes prior knowledge of both Swarm and the basic requirements for a native client communicating with Swarm.

Getting Started

Installation

Install and run the SDK on any operating system supporting the specific programming language being used (Java, Python, C#, or C++).

The Software Development Kit (SDK) distribution ZIP file contains the following:

- Source code for each language implementation (see the language-specific sections for source tree location/contents)
- Language-specific documentation (such as javadocs)
- Language-specific code examples for all SCSP methods and several commonly used query arguments and headers
- Third-party utilities required to compile or run a SDK SCSP client or which provide useful utilities
- LICENSE.txt - Swarm SDK SCSP Client license

Run-Time Configuration

Configuring SDK Timeouts

The SDK enables configuring the following timeouts:

- **ConnectionTimeout:** For Java, Python and C#, sets the length of time an open request socket can be inactive. For C++, the amount of time the client waits for a connection to be opened. C++ also has an additional requestTimeout for the time limit, in secs/GB, for the request to complete. A 2GB write request times out after 400 seconds with a requestTimeout of 200 secs/GB.
- **PoolTimeout:** Sets the amount of time an open connection stays in the connection pool before being closed.
- **LocatorRetryTimeout:** Sets the amount of time between attempts in the locator to retry a node discovered to be unavailable.

Using Connection Pooling

The connection pool stores open, previously used connections for reuse so clients do not need to negotiate opening a socket for every request. The following table shows which parameters control connection pooling for each supported programming language. DataCore recommends setting the value of each parameter shown in the table to the number of threads multiplied by the number of Swarm cluster nodes.

Language	Parameter name
C++	maxStoredConnections
C#	maxStoredConnections
Java	maxConnectionPoolSize
Python	maxSavedConnections

 **Important**

Limit the value to 5 times the number of threads for installations with a large number of nodes and a high thread count (approximately 100 or more) to avoid reaching the client's operating system limits on open file descriptors.

Using Static Location and Locator Types

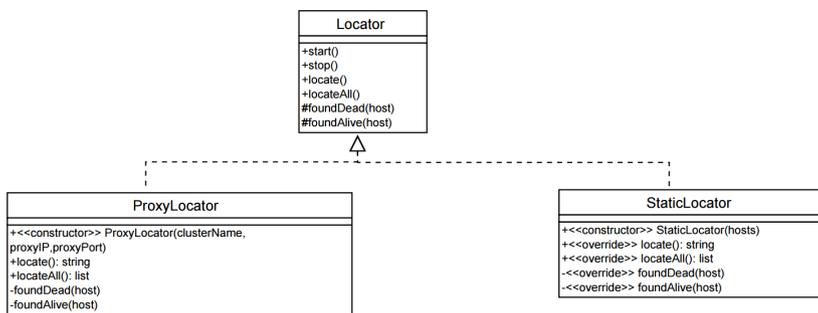
There is a new ProxyLocator subclass available in each SDK programming language that performs the following functions:

1. Performs a GET / to the SCSP Proxy to prepopulate a local list of Swarm node IP addresses.
2. Dynamically maintains this list as redirects and other responses are received directly from Swarm nodes.

The ProxyLocator class API is basically identical to the existing StaticLocator, with the exception that the constructor accepts the following additional parameters:

- SCSP Proxy IP address or host name
- SCSP Proxy's port
- Swarm cluster name

The following UML diagram shows the API of the Locator interface and subclasses, ProxyLocator and StaticLocator:



The locate() method throws an exception including reason field of the Scsp-Proxy-Nodes response header if the SDK node list is currently empty. For more information about Scsp-Proxy-Nodes, see the SCSP Proxy Overview Guide.

Using the HostHeaderValue Property

Supply a HostHeaderValue property on ScspClient overriding the Host header in the request to access a named object. HostHeaderValue specifies the object's

domain name.

ScspClient sends an ScspHeader argument to Swarm using a Host header if an ScspClient method includes it.

ScspClient uses HostHeaderValue property for the Host header if the Host header is empty but the HostHeaderValue is not empty.

ScspClient uses the host in the request URL if both are empty.

Using the Path Argument

The Path argument has changed in SDK version 1.2 to support named objects, unnamed objects, and SCSP Proxy paths.

The following syntax is valid with the SCSP Proxy; sending requests formatted as follows directly to a Swarm cluster results in a 404 (Not Found) error because Swarm attempts to resolve it as a named object.

`/_proxy` is a required prefix for accessing objects using the SCSP Proxy.



Note

The **uuid** parameter is deprecated and replaced by **path** in this release. The **uuid** parameter is planned for removal in a future release.

Syntax	Description
<code>/_proxy/cluster-name/uuid-or-name</code>	Sends a request for an object, referenced by UUID or by name, to a specific cluster name .

<code>/_proxy/all/uuid-or-name</code>	Queries all configured clusters (remote and local) for a particular object, determines the current version, and returns the object to the client.
<code>/_proxy/any/uuid-or-name</code>	<p><code>any</code> is valid for remote INFO and results in an error if used with any other method.</p> <p><code>any</code> causes a request to be sent for an object, referenced by UUID or by name, to any available cluster (local or remote).</p> <p>The information is returned if the object exists in the local cluster. The request is sent to each remote cluster in random order if this condition is not met. The error response from the local server is returned if no cluster is able to locate the data.</p>
<code>/_proxy/remote/uuid-or-name</code>	<p><code>remote</code> is valid for remote INFO and results in an error if used with any other method.</p> <p><code>remote</code> causes a request for an object to be sent, referenced by UUID or by name, to a remote cluster.</p> <p>The information is returned from the first cluster with the object. The error response received from the first remote cluster attempted is returned if the object cannot be found in the remote clusters.</p>

 **Deprecated**

Support for remote SCSP Proxy requests without the `/_proxy` prefix is deprecated and planned for removal in a future release.

Logging

The SDK does not currently provide a standard logging mechanism. Some implementation libraries, such as the HttpClient package on which the java SDK is built, have built-in logging providing useful information. Where applicable, the language implementations provide examples of how to turn on the built-in logging for advanced users.

Errors and Status Codes

The API returns error and status codes or throw exceptions as appropriate to the implementation language. The API throws SCSP-specific exceptions where possible. Certain exception states are challenging to fully anticipate in any language or library.

Common Terminology

Throughout this document and in other documents describing Swarm characteristics, the following terms are used interchangeably:

- **Object** (also referred to as a *stream* or a *file*) is a piece of unstructured content stored in a Swarm cluster.
- **Client** (also referred to as an *application*) is a system or a particular instance of a system that accesses a remote service on a server, Swarm in this instance, by way of a network.

Using SDK Code Examples

All four languages supported by the SDK now offer samples that create, update, and delete objects; and execute other SCSP methods on them.

Verify the following before running these samples:

Tenants

The cluster administrator must create two tenants as follows:

Domain name	Domain protection setting
allusers.realm	All users. No authentication required
localusers.realm	Only users in this domain

In addition, neither domain can have any domain managers.

Contact the cluster administrator to perform these tasks and verify the administrator sets up the tenants exactly as shown in the preceding table.

Credentials

Client applications must specify a valid user name and password for a user in the CAStor administrator realm.

Swarm has a CAStor administrator user named admin with the password `ourpwdofchoicehere` by default. No action is necessary to use these defaults.

Edit the example file before compiling and running it if the the administrator password was changed.

Contact the cluster administrator to get this information.

Local Environment

DataCore recommends running these samples in a non-production environment.

Either edit the Environment.* file for the language being used or set the following local environment variables to set up the proper local environment:

Variable	Meaning
SCSP_HOST	Host name or IP address of a node in the Swarm cluster. This host or IP address must be accessible from the machine where the examples are run.
SCSP_PROXY_HOST	Used by the remote examples. Host name or IP address of the local SCSP Proxy.
SCSP_PORT	Swarm cluster node's SCSP listen port. Default is 80.

ScspClient

ScspClient is a collection of execution classes providing procedural methods for execution of the various SCSP commands. The API supports the following commands:

- Unnamed objects
 - Write, Read, Info, Delete (both mutable and immutable)
 - Update, Copy, Append (mutable)
- Named objects (which are always mutable): Write, Read, Info, Delete, Copy, Update, and Append
- AggregateInfo
- Node Status

In all, Swarm supports the following general types of objects:

- **Immutable unnamed objects**, which can be deleted but not changed. The UUID of an unnamed object is not reused if deleted.
- **Mutable unnamed objects** (anchor streams), which have replaceable contents but UUIDs that do not change. Anchor streams must have the ?alias=yes query argument. Like immutable unnamed objects, an anchor stream's UUID is not reused after the object is deleted.
- **Named objects**, which are mutable but which are addressed by name instead of by UUID. Another named object with the same name can be created later if deleted.

Methods in the ScspClient class now support the value UNDETERMINED_LENGTH for an object's inputStreamLength to support the new large objects feature in Swarm version 6.0. UNDETERMINED_LENGTH can be used with an object (such as a live video feed) whose size is not known and causes an object to be sent to Swarm using standard [HTTP chunked transfer coding](#).

In addition, all SDK languages support [query arguments](#) for erasure-coded objects: ?checkIntegrity, ?encoding, and ?segmentSize.

See [Erasure Coding EC](#).

SCSP Classes

The following sections briefly discuss the SCSP operations supported by the SDK in all languages. See the language-specific sections for additional operations that may be supported.

Write	The Write method writes an object to the specified cluster and returns a name or UUID. It is equivalent to an HTTP POST. A successful Write method execution returns an HTTP response code of either 201 or 202.
Read	The Read method returns an object requested by name or UUID from a specified cluster. It is equivalent to an HTTP GET. A successful Read method execution returns an HTTP response code of 200. A successful Read with range headers may return a 206 response code. A successful Read with cache coherency (for example, if-match) headers may return a 304 Not Modified response code.

ScscpClient	
Delete	The Delete method deletes (if policy allows) an object identified by name or UUID from a specified cluster. It is equivalent to an HTTP DELETE. A successful Delete method execution returns an HTTP response code of 200.
Info	The Info method returns the metadata for an object identified by name or UUID from a specified cluster. It is equivalent to an HTTP HEAD. A successful Info method execution returns an HTTP response code of 200.
AggregateInfo	The AggregateInfo method is an extension of the normal SCSP operations. It must be used in requests to the SCSP Proxy; it cannot be used in requests directly to Swarm nodes. In a single request, get a list of names or UUIDs to be Info'd to the specified cluster using the following format: A list of either URL-encoded names or UUIDs must be supplied. (Use percent encoding for object names, if needed.) The mutable parameter must be used for unnamed anchor streams. (The default is immutable.) The name, UUID, or consistency checkpoint is stored as a Swarm object. Then in the AggregateInfo method, Info requests are issued for each name or UUID in the consistency checkpoint and either object metadata or an error response is returned in the concatenated response body. Similar to the individual Info method, the response for a successful AggregateInfo method execution is a 200 code. The AggregateInfo method supports named and unnamed objects for both the manifests and streams stored in the manifest. It supports authentication for the manifest stream itself, and checkpoint streams in the manifests protected for HEAD are returned as 401s in the AggregateInfo response body.
NodeStatus	The NodeStatus method returns basic capacity information for the cluster as well as some high level SCSP operations counts for the queried node in the response body. The total and available capacity numbers are also returned as headers on the response, Castor-System-TotalGBCapacity and Castor-System-TotalGBAvailable respectively. In addition to the status data, this method can be useful in verifying an SDK client is talking to a live Swarm node. A successful NodeInfo method execution returns an HTTP response code of 200.
WriteMutable	The WriteMutable method writes a mutable object to the specified cluster and returns a name or UUID. It is equivalent to an HTTP POST with an alias=yes query argument for anchor streams. A successful WriteMutable method execution returns an HTTP response code of either 201 or 202.
<pre> +Connection Pool +UserAgent: String +ScscpClient(Hosts:String,Port:Integer, MaxConnections:Integer,Retries:Integer,ConnectionTimeout:Integer, PoolTimeout:Integer):ScscpClient +ScscpClient(Locator:Locator,Port:Integer, MaxConnectionPoolSize:Integer, Retries:Integer,ConnectionTimeout:Integer, PoolTimeout:Integer):ScscpClient +Start() +Stop() +CreateWriteCommand(uuid:UUID,Input:Input,InputLength:Long, queryArgs:ScscpQueryArgs):ScscpWriteCommand +CreateReadCommand(uuid:UUID,Output):ScscpReadCommand +CreateDeleteCommand(uuid:UUID):ScscpDeleteCommand +CreateInfoCommand(uuid:UUID,Output):ScscpInfoCommand +CreateAggregateInfoCommand(uuid:UUID,Output):ScscpAggregateInfoCommand +CreateUpdateCommand(uuid:UUID,Input,InputLength:Long):ScscpUpdateCommand +CreateAppendCommand(uuid:UUID,Input,InputLength:Long):ScscpAppendCommand +CreateCopyCommand(uuid:UUID):ScscpCopyCommand +Write(path:String,Input,InputLength:Long, queryArgs:ScscpQueryArgs):ScscpResponse +Read(uuid:UUID,path:String,Output,queryArgs:ScscpQueryArgs, metaData:ScscpHeaders):ScscpResponse +Delete(uuid:UUID,path:String,queryArgs:ScscpQueryArgs, metaData:ScscpHeaders):ScscpResponse +Info(uuid:UUID,path:String,queryArgs:ScscpQueryArgs, metaData:ScscpHeaders):ScscpResponse +AggregateInfo(uuid:UUID,path:String,queryArgs:ScscpQueryArgs, metaData:ScscpHeaders):ScscpResponse +NodeStatus(queryArgs:ScscpQueryArgs):ScscpResponse +WriteMutable(path:String,Input,InputLength:Long, queryArgs:ScscpQueryArgs):ScscpResponse +ReadMutable(uuid:UUID,path:String,Output, queryArgs:ScscpQueryArgs,metaData:ScscpHeaders):ScscpResponse +UpdateMutable(uuid:UUID,Input,InputLength:Long,queryArgs:ScscpQueryArgs, metaData:ScscpHeaders):ScscpResponse +DeleteMutable(uuid:UUID,path:String,queryArgs:ScscpQueryArgs, metaData:ScscpHeaders):ScscpResponse +InfoMutable(uuid:UUID,path:String,queryArgs:ScscpQueryArgs, metaData:ScscpHeaders):ScscpResponse +AppendMutable(uuid:UUID,Input,InputLength:Long,queryArgs:ScscpQueryArgs, metaData:ScscpHeaders):ScscpResponse +CopyMutable(uuid:UUID,path:String,queryArgs:ScscpQueryArgs, metaData:ScscpHeaders):ScscpResponse +ValidateMode(on:Bool) </pre>	
<p>Note</p> <p>Anchor streams are not intended to be used as general-purpose, updateable objects. In particular, rapid reads or updates (more than once per second) of an anchor stream can produce unpredictable results and may result in errors being logged or error responses being returned to the application.</p>	

ReadMutable	The ReadMutable method returns a mutable object requested by name or UUID from the specified cluster. It is equivalent to an HTTP GET with an alias=yes query argument for anchor streams. A successful ReadMutable method execution returns an HTTP response code of 200 or 206.
UpdateMutable	The UpdateMutable method updates an existing mutable object in the specified cluster. It is equivalent to an HTTP PUT. A successful UpdateMutable method execution returns an HTTP response code of either 201 or 202.
DeleteMutable	The DeleteMutable method deletes (if policy allows) a mutable object identified by name or UUID from a specified cluster. It is equivalent to an HTTP DELETE with an alias=yes query argument for unnamed anchor streams. A successful DeleteMutable method execution returns an HTTP response code of 200.

InfoMutable	The InfoMutable method returns the metadata for a mutable object identified by name or UUID in the specified cluster. It is equivalent to an HTTP HEAD with an alias=yes query argument for unnamed anchor streams. A successful InfoMutable method execution returns an HTTP response code of 200.
AppendMutable	The AppendMutable method allows appending of new data on to the end of the content for an existing mutable object in the specified cluster. There is no HTTP equivalent for Append. A successful AppendMutable method execution returns an HTTP response code of either 201 or 202.
CopyMutable	The CopyMutable method allows metadata update without modifying the content of an existing mutable object in the specified cluster. There is no HTTP equivalent for Copy. A successful CopyMutable method execution returns an HTTP response code of 201 or 202.

Support Classes

Support classes are also available to implement common functions for ScspClient. The following support classes provide building blocks for creation of some of the components needed for various commands:

ScspIntegritySeal	This class provides integrity seal hash types for setting up write headers as well as integrity seal response parsing.
ScspAuthentication	This class provides a method to return a realm in response headers and to send a user name and password with request headers.
ScspAuthorization	Enables usage of authentication with Swarm 5.0 and later
ScspDate	This class provides the basic functionality for converting to and from language-specific dates to SCSP date formats, for both query arguments and Lifepoints.
ScspLifepoint	This class provides the components for building a single Lifepoint.
ScspDeleteConstraint	This class enumerates the standard Lifepoint delete constraint strings: deletable=no, deletable=yes, and delete.

See the language-specific sections ([SDK for C#](#), [SDK for C++](#), [SDK for Java](#), [SDK for Python](#)) for specifics on how each language implements the support classes.

Validation Mode

ScspClient can be run in validation mode to verify proper formatting of some of the most commonly used query arguments and headers. The SDK can be run in Validation mode without being connected to any SCSP server or SCSP Proxy, in which case the following is validated.

Validation	Read	Info	Write	Delete	Update	Copy	Append
Header: Allow			V		V	V	
Header: Content- Type			V		V	V	
Header: Content- Disposition			V		V	V	
Header: Host	V	V	V	V	V	V	V
Query argument: replicate			V		V	V	V
Query argument: alias	V	V	V	V	V	V	V
Query argument: validate	V						
Query argument: hashtype	V		V		V	V	V

Query argument: hash	V						
Query argument: newhashtype	V						
Query argument: countreps		V					
Query argument: domain	V	V	V	V	V	V	V
Query argument: admin	V	V	V	V	V	V	V

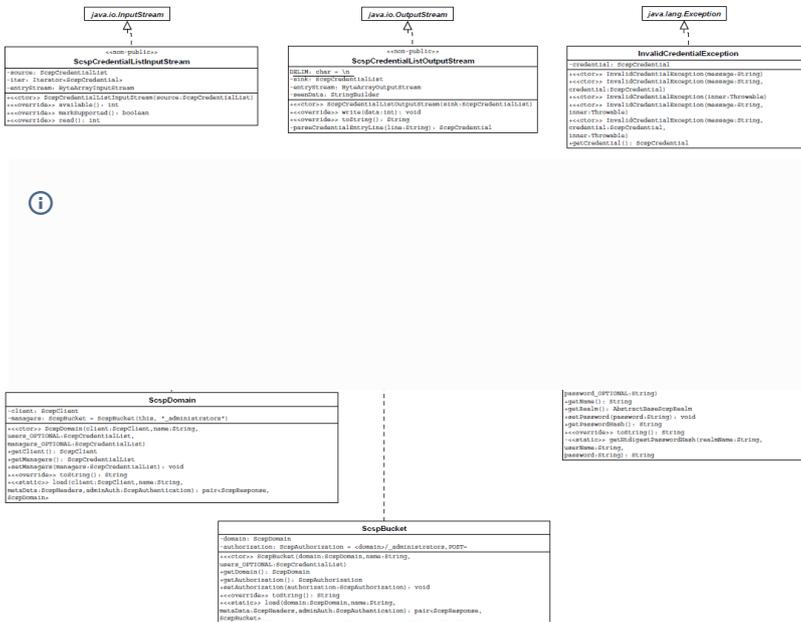
Managing Domains and Buckets

The SDK enables adding, renaming, or deleting domains to assist in managing domains (also referred to as tenants) and buckets. For more information about named objects and tenancy, see the Swarm Application Guide.

The source code referred to in this appendix is located as follows:

- C++: sdk-extract-dir /cpp/src/realm
- C#: sdk-extract-dir \csharp\ScspCSEExamples\ScspRealmExamples.cs
- Java: CASTorSDK-src-extract-dir /com/caringo/realm
- Python: castorsdk-python-egg-extract-dir /castorsdk/realm

Refer to the following UML diagram:



The SDK provides the following classes:

- **ScspDomain**, which creates, modifies, and deletes the domain object.

Note

Execute the methods in ScspDomain using a single thread to avoid having the same domain created by more than one administrator.

- **ScspBucket**, which creates buckets. ScspBucket has an attribute called (get/set) authorization controlling the Castor- Authorization header for the bucket.
- **ScspCredential**, which holds the authorization specification for a specific user.
- **ScspCredentialList**, which holds the list of credentials defined for a given user list.

- **ScspCredentialListInputStream**, which enables reading the credential list using an InputStream interface.
- **ScspCredentialListOutputStream**, which enables building the credential list using an OutputStream interface.
- **InvalidCredentialException**, which is thrown when an invalid credential has been detected (such as a user list mismatch).

Object Headers and Query Arguments

The basic methods supported by the API can be extended with the addition of both standard HTTP request and Swarm-specific headers and/or query arguments. These optional components are separated from the methods themselves to support custom headers and headers and query arguments to the SCSP protocol over time.

ScspHeaders	ScspQueryArgs
<pre> +HeaderList: Map<String, List<String>> +AddAll(other: ScspHeaders) +AddValue(name: String, value: String) +ReplaceValue(name: String, value: String) +SetValues(name: String, values: List<String>) +Remove(name: String) +RemoveAll() +GetHeaderValues(name: String): List<String> +ContainsName(name: String): Boolean +NameCount(): Integer +HeaderCount(): Integer +AddLifetime(date: ScspDate, deleteConstraint: DeleteConstraint, reps: Integer) +AddRange(start: Long, end: Long) +SetAuthentication(ScspAuthentication): void +GetAuthentication(): ScspAuthentication +ToString(): String +ToHeaderList(): List<ScspHeader> </pre>	<pre> +ArgList: Map<String, String> +SetValue(name: String, value: String) +GetValue(name: String): String +AddAll(newArgs: Map<String, String>) +ContainsName(name: String): Boolean +ToQueryString(): String +Remove(name: String) </pre>

Using ScspHeaders for System and Custom Metadata

About SCSP Headers

In HTTP and SCSP, metadata for requests, responses, and content itself are all represented by line-oriented, textual headers prefixing any binary data included with the message. The ScspHeaders class eases the creation and parsing of common header value syntax, enabling associating any string values with any header name.

All SCSP client language implementations included in this SDK support standard HTTP request headers like Content-Type, Content-Length, Content-MD5, and Content-Disposition. The included clients also enable applications to use Swarm-specific headers, including Lifepoints and Castor-Authorization.

Any number of headers can be created but they are not filtered or validated in any way by default. The SDK validates Content-Type, Content-Disposition, and Allow headers when run in validation mode. Multi-value headers are supported and can be created using either of the standard HTTP mechanisms for defining headers:

- A single header name with multiple string values
- Multiple entries for the same header name with different string values

All language implementations of the SDK, except C#, return headers verbatim from the Swarm response. In C#, because of the underlying HTTP header handling in .NET, the SDK splits header values in to multiple header entries (except for Lifepoints, any headers with a name including 'date', and the Castor-System-Created header).

About the CAStor-system-* Header

Castor-System-* headers are reserved for internal use and are therefore not allowed on an incoming client request. The header is silently ignored if such a header is present in a request.

ScspQueryArgs

HTTP query arguments are key/value pairs passed in the HTTP request along with the URL. Similar to ScspHeaders, the ScspQueryArgs object allows arbitrary association of names and string arguments as well as the following Swarm-specific arguments.

Argument	Description	Commands
alias=yes	Mutable unnamed objects (anchor streams)	Write, Read, Info, Copy, Update, Append
domain=domain-name	Named objects	Write, Read, Info, Copy, Update, Append, Delete

<code>replicate=immediate</code>	Replicate on write	Write, Copy, Update, Append
<code>validate=yes</code>	Check stored entity digest while reading	Read
<code>hashtype=hash-algorithm, hash=digest</code>	Compute and return an integrity seal	Write, Read, Copy, Update, Append
<code>newhashtype=hashalgorithm</code>	Compute and return an integrity seal	Read
<code>countreps=yes</code>	Return the current number of replicas for this name or UUID	Info

In addition, `ScspQueryArgs` allows callers to pass in arbitrary query arguments merged with any automatically generated ones and passed along with the request URL.

See [SCSP Query Arguments](#) for all query arguments possible for each SCSP command.

SDK for C#

- [C# SDK Installation and Packaging](#)
- [Required Environment](#)
- [Implementation Notes](#)
- [Using the C# Client Sample Code](#)
- [C# Administrative Override of an Allow Header](#)

C# SDK Installation and Packaging

The C# distribution contains the following directory structure:

- `ScspCS`
 - `Properties`: the properties of the Visual Studio project file
 - Visual Studio project (`ScspCS.csproj`)
 - source files: `*.cs` files for each of the major components
- `ScspCSExamples`: code examples for all major functions

Required Environment

The C# client requires the following libraries and utilities:

1. Microsoft Visual Studio C# 2013 or Visual Studio C# Express 2013 with the NuGet package manager enabled
2. .NET 4.5 or later

Implementation Notes

Using Connection Pooling with the C# Client

The connection pool stores open, previously used connections for reuse so the client does not need to negotiate opening a socket for every request. You configure connection pooling for the C# client using the `maxStoredConnections` parameter in the `ScspClient` class. For best results, set the value of this parameter to the number of threads multiplied by the number of Swarm cluster nodes.



Best practice

For installations with a large number of nodes and a high thread count (approximately 100 or more), limit the value to 5 times the number of threads to avoid reaching the client's operating system limits on open file descriptors.

`ScspClient.Close` is required and does not serve as a no-op.

Notes About the C# SDK

Following are notes to when writing C# code for the SDK:

- **C# Write, Update, Append.** A Write, Update, or Append using the C# SDK client that encounters an error response, an `ScspWebException` may be thrown. This can occur with a 400 response from the cluster, or on any error response (code 400 and greater) when using the SCSP Proxy. This behavior is caused by the way that .NET internally handles a connection closing while writing data to a peer. There is no known workaround.
- **Connection timeout.** The connection timeout provided to `ScspClient` is used for two purposes: First as a timeout for connecting and individual read/write API calls, and secondly as a basis for a timeout for the overall HTTP request (this is due to .NET behavior; see [HttpWebRequest.Timeout](#) for details). You can also use the size of the object being uploaded as a guide for setting the overall timeout for the entire request. For example, if the connection timeout is set to 300 seconds, a 2GB write has a full request timeout of approximately 600 seconds. Increasing the connection timeout may help alleviate write failures on large objects due to too many retries of cancelled requests.
- **.NET support.** .NET 4.5 is required in versions 6.1.1 and later of the SDK. Applications using an older version of .NET must install .NET 4.5 prior to upgrading to version 6.1.1 or later.
- **Character encoding.** If you pass in a URI path, you must escape a backslash character (\) with %5c.
- **Range.** You can now use 64-bit Ranges with the C# SDK when using .NET 4.5.

Using the C# Client Sample Code

The C# SDK client ships with runnable sample code that demonstrates how to write client applications for Swarm. The C# sample code is located in the `sdk-zip-extract-dir/caringo-sdk-version-brand/csharp/ScspCSEExamples` directory and consists of the following files:

C++ sample code file name	Description
<code>ScspExample.cs</code>	General SCSP client examples.
<code>ScspRemoteExample.cs</code>	Examples of performing SCSP operations on local and remote clusters using the SCSP Proxy.

Exploring the C# Sample Code

The `sdk-extract-dir/caringo-sdk-version-brand/doc/examples` directory contains commented C# sample code you can view in a web browser. Double-click `index-all.html` to open the index page in your default web browser. The commented samples include the following:

- Authenticating named objects
- Performing SCSP operations on unnamed mutable and immutable objects
- Using lifepoints and MD5 integrity seals
- Using READ, including validation, ranges, and Content-MD5
- Performing SCSP operations on objects in local and remote clusters using the SCSP Proxy

C# Administrative Override of an Allow Header

This example is not in the sample code provided with the SDK.

```
// This example shows how to execute an update command
// with administrative override.
String DEFAULTADMINREALM = "CASTor administrator";

ScspUpdate uc = client.CreateUpdateCommand(uuid, inputStream, testDataLength);
uc.UserAgent = ScspClient.CASTOR_ADMIN_AGENT;
ScspAuthentication auth = new ScspAuthentication("admin",
    "ourpwdofchoicehere", DEFAULTADMINREALM);
headers.Authentication = auth;
uc.Headers = headers;
response = uc.Execute();
```


SDK for Python

- [Installing the Python Client](#)
- [Client Startup Behavior](#)
- [Using Connection Pooling with the Python Client](#)
- [Implementation Notes](#)
- [Using the Python Client Sample Code](#)
- [Python Administrative Override of an Allow Header](#)

Installing the Python Client

The following prerequisites must be installed Before installing the Python client:

- Python 2.5 or 2.6
DataCore has tested the Python SDK with the preceding versions of Python. Using other versions may have unpredictable results.
- [Python setuptools package](#) for the version of Python being used

Installing the Python SDK on Python 2.5 or 2.6

The Python client is packaged in an easy to install egg file (python castorsdk-version- pyversion.egg) that contains both source and compiled code. To install the egg on a supported version of Python, enter the following command as a user with root privileges:

```
easy_install castorsdk-version-pyversion.egg
```

The easy install may print some errors related to not finding an index page or not finding a suitable distribution path. These errors can be safely ignored.

Client Startup Behavior

ScspClient.start() is an empty (pass) method. Client startup is implicit in the execution of a request using an ScspClient execution method. This behavior is different from other language implementations of the SDK, which fail when a request is issued after a stop. It is still important that the SDK Python client call ScspClient.stop() at the end of execution so the SDK can clean up any cached open connections.

Using Connection Pooling with the Python Client

The connection pool stores open, previously used connections for reuse so a client does not need to negotiate opening a socket for every request. Configure connection pooling for the Python client using the maxSavedConnectionsparameter in the ScspClient class. DataCore recommends setting the value of this parameter to the number of threads multiplied by the number of Swarm cluster nodes.



Note

Limit the value to 5 times the number of threads to avoid reaching the client's operating system limits on open file descriptors for installations with a large number of nodes and a high thread count (approximately 100 or more).

Implementation Notes

- **Character encoding.** a backslash character (\) must be escaped with %5c if passing in a URI path.
- **ScspiOError exception class.** There is an exception class (ScspiOError, derives from IOError) that bypasses the normal retry logic when there is an error reading body data in SDK version 6.0 and later. This class throws an error anytime an object read is attempted but cannot be completed.

Using the Python Client Sample Code

The Python SDK client ships with runnable sample code that demonstrates how to write client applications for Swarm. The Python sample code is located inside the egg file. The examples are located in the `sdk-zip-extract-dir/caringo-sdk-version-brand/python/castorsdk/examples` directory after expanding the egg file and consist of the following files:

Python sample code file name	Description
Environment.py[c]	Sets the environment to run all SDK code examples.
ScspExample.py[c]	General client examples.
SCSPRemoteExample.py[c]	Client examples that use the SCSP Proxy with local and remote clusters.
ObjectEnumerator- Example.py[c]	ObjectEnumerator class examples for use with the Content Router. Content Router is deprecated and these examples are removed in a future release.
RealmExamples.py[c]	Example of creating a domain and bucket.

Exploring the Python Sample Code

The `sdk-extract-dir/SwarmSDK-version/docs/example` directory contains commented Python sample code viewable in a web browser. Double-click `index-all.html` to open the index page in the default web browser. The commented samples include the following:

- Authenticating named objects
- Performing SCSP operations on unnamed mutable and immutable objects
- Using lifepoints and MD5 integrity seals
- Using READ, including validation, ranges, and Content-MD5
- Performing SCSP operations on objects in local and remote clusters using the SCSP Proxy

Creating domains and buckets

Preparing to Run the Python Code Examples

To get started:

1. Extract the SDK .zip file into an empty directory if not already done so. This directory is referred to as `sdk-extract-dir`.
2. Extract `castorsdk-version-pyversion.egg` into the following directory if not already done so, :
`sdk-extract-dir/python`
3. Open a command prompt window.
4. Set the local environment by either editing the following file or setting local environment variables SCSP_HOST, SCSP_PROXY_HOST, SCSP_PORT, PUBLISHER_HOST, and PUBLISHER_PORT:

```
Python-source-code-extract-dir/python/castorsdk/examples/environment.py[c]
```

5. The example file must be edited before compiling and running it if the cluster administrator changed the default CASTor administrator password. The default password is ourpwdofchoicehere. Search for that string in the example files and change it before compiling and running the example.

Running the Python Examples

This section discusses how to run the Python examples after compiling them as discussed in the preceding section.

1. Set a local environment variable PYTHONPATH to this directory:
`sdk-extract-dir/python/castorsdk`
2. Change to the `sdk-extract-dir/python/castorsdk` directory and enter the following command to run all examples except the multi-tenancy examples:

```
python examples/scspExamples.py
```

3. Run the remote cluster examples as follows:

```
python examples/scspRemoteExamples.py
```

4. Set a local environment variable PYTHONPATH to the `sdk-extract-dir/python` directory.
5. Change to the `sdk-extract-dir/python` directory and enter the following command to run the multi-tenancy examples:

```
python castorsdk/examples/realExamples.py
```

Basic Example Troubleshooting

Use the following tips to troubleshoot any errors encountered running the examples:

Some scspExamples are expected to fail.

The first lifepoint example creates an object with an immediate delete, so subsequent methods on that object fail. An example follows:

```
>>>> LP Example <<<<<<
Set a terminal lifepoint to delete. Note that the stream will be deleted as soon as it's written.
...[commands omitted]
>>>>> Read <<<<<<
Status Line HTTP/1.1 404 Not Found Status Code 404
Headers
Content-Length: 92
Content-Type: text/html
Date: Sat, 03 Dec 2011 00:00:54 GMT
Server: CASTor Cluster/6.0.0
Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE

Response Body HTTP/1.1 404 Not Found
Content-Length: 92 Content-Type: text/html
Date: Sat, 03 Dec 2011 00:00:54 GMT
Server: CASTor Cluster/6.0.0
Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE

<html><body><h2>CASTor Error</h2><br>Requested stream was not found (ENOENT)</body></html>

Result Code 0
Retry Count 0
Root error None
```

412 (Precondition Failed)

412 (Precondition Failed) is expected in some SCSPClientExample ETag examples, such as the following:

```
>>>>> Read Etag <<<<<<

Status Line HTTP/1.1 412 Precondition Failed Status Code 412
Headers
Last-Modified: Sat, 03 Dec 2011 00:00:56 GMT
Etag: "668897166a53016eb8b3792a0d12a87d"
Content-Length: 77
Content-Type: text/html
Date: Sat, 03 Dec 2011 00:00:56 GMT
Server: CAStor Cluster/6.0.0
Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE
Connection: close

Response Body HTTP/1.1 412 Precondition Failed
Last-Modified: Sat, 03 Dec 2011 00:00:56 GMT
Etag: "668897166a53016eb8b3792a0d12a87d"
Content-Length: 77
Content-Type: text/html
Date: Sat, 03 Dec 2011 00:00:56 GMT
Server: CAStor Cluster/6.0.0
Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE
Connection: close
<html><body><h2>CAStor Error</h2><br>Stream has been modified</body></html>

Result Code 0
Retry Count 0
Root error None
```

This example fails because the object was changed by the sample code.

401 (Unauthorized)

401 (Unauthorized) is expected in some SCSPClientExample authentication examples, such as the following:

```
>>>>> Info named object 'protectedobject' without credentials<<<<<< Status Line HTTP/1.1 401 Una
Status Code 401
Headers
Content-Length: 0
WWW-Authenticate: Digest realm="allusers.realm/2814339560", nonce="2a05cf4e8ff625f5e06bf12aefb0ce
opaque="97024db82e5e3f455e845cb89ee89e08", stale=false, qop="auth", algorithm=MD5
WWW-Authenticate: Basic realm="allusers.realm/2814339560"
Date: Sat, 03 Dec 2011 00:01:51 GMT
Server: CAStor Cluster/6.0.0
Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE

Response Body HTTP/1.1 401 Unauthorized
Content-Length: 0
WWW-Authenticate: Digest realm="allusers.realm/2814339560", nonce="2a05cf4e8ff625f5e06bf12aefb0ce
opaque="97024db82e5e3f455e845cb89ee89e08", stale=false, qop="auth", algorithm=MD5
WWW-Authenticate: Basic realm="allusers.realm/2814339560"
Date: Sat, 03 Dec 2011 00:01:51 GMT
Server: CAStor Cluster/6.0.0
Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE

Result Code 0
Retry Count 0
Root error None
```

This failure is expected because the method requires authentication but no credentials are supplied.

Domains

scspExamples that depend on domains being set up fail if the domains are not set up properly. Examples follow:

```
Realm 'allusers.realm' doesn't appear to have been set up correctly:
Castor-Authorization header doesn't look right. Returning.
```

The preceding error indicates the allusers.realm was set up with the wrong domain protection setting.

```
Couldn't find realm 'localusers.realm'. Returning.
```

The preceding error indicates the realm does not exist.

Review the information discussed in [SDK Overview](#) and attempt running the examples again.

SCSP Proxy

Python exceptions display, including the following in the event the SCSP Proxy or Content Router are not set up properly or are unavailable:

```
Can't write to proxy host. It looks like I can't talk to your proxy or its remote server.
Please check your configuration and retry.
```

Python Administrative Override of an Allow Header

This example is not included in the code samples provided with the SDK.

```
# This example shows how to execute a copy command with administrative
# override to clean up Allow headers. Note that you may want to get the
# original headers first to make sure none of the existing ones on the stream
# are lost.
```

```
DEFAULTADMINREALM = 'CASTor administrator'

#set admin credentials
auth = ScspAuthentication()
auth.realm = DEFAULTADMINREALM #realm doesn't matter
auth.cnonce = 'abcdef' # actual value doesn't matter
auth.user = 'admin'
auth.password = 'ourpwdofchoicehere'
headers = ScspHeaders()
headers.addValue('Allow', 'PUT')
headers.authentication = auth
args = ScspQueryArgs()
#make this request administrative
client.copy(uuid, args, headers)
```

SDK for Java

- [Source Directory Structure](#)
- [Java Client Implementation Notes](#)
- [Using the Java Client Sample Code](#)
- [Java Administrative Override of an Allow Header](#)

Source Directory Structure

The Java source is organized as follows in CASTorSDK-src.zip:

- with
 - `caringo`
 - `client`: Main SCSP Client files, including subfolders
 - `examples`: File(s) showing how to use the Scsp Client to perform various SCSP actions.
 - `locate`: Files used to help SCSP clients locate Swarm instances. The ScspClient class uses either ProxyLocator or StaticLocator to track the configured list of hosts. RoundRobinDnsLocator is a base class for the other locator classes. It implements a Locator based on Round-robin DNS. These examples are unsupported and untested.
 - `request`: Files implementing the Java SCSP communication engine.
 - `examples/config` has `log4j.properties` that shows how to use HttpClient log4j logging

Java Client Implementation Notes

Building the Java SDK

The source zip includes a Maven project file for building the Java SDK. The CastorSDK.jar was built using Maven 3.2.5 and Java 7 (jdk 1.7.0_79). The build depends on the following Maven projects:

- testng 6.8
- Apache httpcomponents.httpclient 4.2.5 and httpcomponents 4.4.4. (Although httpclient 4.5.3 is released, it has not been validated with the SDK.)
- Apache log4j 1.2.17
- Maven javax.jmdns 3.4.1

To build the uber jar from scratch, run the following at a command prompt:

```
mvn clean package
```

Java Client Recompile Required

The Java client code using the classes provided with the SDK must be recompiled when upgrading to version 6.1.4 or later because of internal changes made to the Java SDK client.

Using Connection Pooling with the Java Client

The connection pool stores open and previously used connections for reuse so the client does not need to negotiate opening a socket for every request. Configure connection pooling for the Java client using the `maxConnectionPoolSize` parameter in the `ScspClient` class. DataCore recommends setting the value of this parameter to the number of threads multiplied by the number of Swarm cluster nodes.

For more information about this parameter, see the **Class `ScspClient`** topic in the Javadoc provided with the SDK, located in the `/java/CASorSDK-doc.zip` file in the SDK package.



Note

Limit the value to 5 times the number of threads to avoid reaching the client's operating system limits on open file descriptors for installations with a large number of nodes and a high thread count (approximately 100 or more).

Remote Cluster Replication and Remote Synchronous Write

The SCSP storage API provides a `SEND` method that can be issued against a source cluster to orchestrate the transfer of a single stream to a destination cluster to support movement of objects between multiple Swarm storage clusters. SCSP also supports a special type of `GET` request called a `GET/retrieve` that triggers a destination cluster to retrieve an object from a source cluster. The Java SDK includes several classes and methods that utilize these SCSP methods to facilitate movement of objects between clusters. The `Replicator` class includes methods for `SEND`ing a single stream, for performing a `GET/retrieve` on a single stream, and for replicating a single stream using `SEND` or `GET/retrieve` via the single-stream methods. It also includes a `Remote Synchronous Write (RSW)` method to write and then immediately replicate a single stream.

When choosing which methods to use for remote replication, the following guidance applies:

- `SEND` is less complex and faster but requires all intervening network components (proxies, gateways, firewalls, routers, etc.) must pass through the `SEND` method, which is a non-standard extension of HTTP 1.1. Network components not recognizing the method may return an `HTTP 501 Not Implemented` response. `GET/retrieve` needs to be used instead.
 - For the lifetime of a `Replicator` instance, the `remoteSynchronousWrite` method attempts to use the `SEND` method, falling back to `GET/retrieve` if `SEND` is not supported.
- `SEND` requires the client have a path to the local cluster. `GET/retrieve` may require a network route to both the source and the destination clusters.

Notes About the Java SDK

- **Error Returned for Locator Empty IP Address** Passing an empty IP Address to the `ProxyLocator` constructor on a RHEL platform fails with a 'Too many SCSP retries' exception instead of the expected `IllegalArgumentException`. The same behavior returns the expected exception on a Windows platform.
- **Character encoding.** Escape a backslash character (`\`) with `%5c` if passing in a URI path.
- **Maximum Open Connections** The `maxConnection` parameter in the Java implementation of the SDK configures the maximum total open connections, which is the sum of both stored and running connections.
- **ResettableFileInputStream** `FileInputStream()` cannot be used; instead, use `ResettableFileInputStream`, which is located in `java\com\caringo\client`.
- **InputStreams** must support `reset()`; (`markSupported()` must return true).
- **OutputStream Cannot be Null** On reads, the value provided for `OutputStream` cannot be null. An `IllegalArgumentException` is thrown if null is provided.

Using the Java Client Sample Code

The Java SDK client ships with runnable sample code that demonstrates how to write client applications for Swarm. The Java sample code is located in subdirectories of `sdk-zip-extract-dir/caringo-sdk-version-brand/java/com/caringo`:

Subdirectory	Java sample code file name	Description
client/examples	<ul style="list-style-type: none"> Environment.java ScspExample.java SCSPRemote-Example.java 	<ul style="list-style-type: none"> Sets the environment to run all SDK code examples. General client examples. Client examples that use the SCSP Proxy with local and remote clusters.
realm/examples	RealmExamples.java	Example of creating a domain and bucket.

Exploring the Java Sample Code

The `sdk-extract-dir/caringo-sdk-version-brand/doc/examples` directory contains commented Java sample code viewable in a web browser. Double-click `index-all.html` to open the index page in the default web browser. The commented samples include the following:

- Authenticating named objects
- Performing SCSP operations on unnamed mutable and immutable objects
- Using lifepoints and MD5 integrity seals
- Using READ, including validation, ranges, and Content-MD5
- Performing SCSP operations on objects in local and remote clusters using the SCSP Proxy
- Creating domains and buckets

Compiling and Running the Java Code Examples

Complete the tasks discussed in “Using SDK Code Examples” in the [SDK Overview](#) before continuing.

To get started:

1. Extract the SDK .zip file into an empty directory if not already done so. This directory is referred to as `sdk-extract-dir`.
2. Extract the Java source into the following directory if not already done so:
`sdk-extract-dir/java`
3. Open a command prompt window.
4. Set the local environment by either editing the following file or setting local environment variables SCSP_HOST, SCSP_PROXY_HOST, SCSP_PORT, PUBLISHER_HOST, and PUBLISHER_PORT:
`java-source-code-extract-dir/com/caringo/client/examples/Environment.java`
5. The example file must be edited before compiling and running it if the cluster administrator changed the default CASTor administrator password. The default password is `ourpwdofchoicehere`. Search for that string in the example files and change it before compiling and running the example.
6. To make setting the classpath easier, set a local environment variable BASE_DIR to the following directory:
`sdk-extract-dir/java`

Compiling the Java Examples

This section discusses how to compile all Java examples. Compile the examples intended to run.

Compile the Java examples:

1. Add the uber-jar to the classpath, using either a local environment variable or the `-classpath classpath` option on the `javac` command line:

```
$BASE_DIR
```

2. Compile it as follows if setting the environment by editing `Environment.java`:

```
javac Environment.java
```

3. Change to the `sdk-extract-dir/java/com/caringo/client/examples` directory and compile `SCSPClientExample.java` as follows to compile the SCSP client and remote examples:

```
javac SCSPClientExample.java
```

4. Compile `SCSPRemoteExample.java` as follows:

```
javac SCSPRemoteExample.java
```

5. Change to the `sdk-extract-dir/java/com/caringo/realm/examples` directory and compile `RealmExamples.java` as follows to compile the multi-tenancy examples:

```
javac RealmExamples.java
```

Running the Java Examples

This section discusses how to run the Java examples after compiling them as discussed in the preceding section.

1. change to the `sdk-extract-dir/java/com/caringo/client/examples` directory and enter the following command to run the SCSP client and remote examples:

```
java com.caringo.client.examples.SCSPClientExample
```

2. Run `SCSPRemoteTest.java` as follows:

```
java com.caringo.client.examples.SCSPRemoteExample
```

3. Change to the `sdk-extract-dir/java/com/caringo/realm/examples` directory and enter the following command to run the multi-tenancy examples:

```
java com.caringo.realm.examples.RealmExamples
```

Basic Example Troubleshooting

Use the following tips to troubleshoot any errors encountered running the examples:

Some `SCSPClientExample` examples are expected to fail

The first lifepoint example creates an object with an immediate delete so subsequent methods on that object fail. An example follows:

```
>>>> LP Example <<<<<<
Set a terminal lifepoint to delete. Note that the stream will be deleted as soon as it's written.
LP: [] delete
...[commands omitted] >>>>> Read <<<<<<<<
HTTP/1.1 404 Not Found
Date: Fri, 02 Dec 2011 15:49:38 GMT
Content-Length: 92
Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE
Content-Type: text/html
Server: CASTor Cluster/6.0.0
<html><body><h2>CASTor Error</h2><br>Requested stream was not found (ENOENT)</body></html>

Retries: 0
ResultCode: ScspRCFailure
```

412 (Precondition Failed)

i412 (Precondition Failed) is expected in some SCSPClientExample ETag examples, such as the following:

```
>>>>> Read ETAG <<<<<<<<
HTTP/1.1 412 Precondition Failed
Date: Fri, 02 Dec 2011 15:49:39 GMT
Content-Length: 77 Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE
Last-Modified: Fri, 02 Dec 2011 15:49:39 GMT
Etag: "75cfb22dd55b800238367a905c927040"
Connection: close
Content-Type: text/html Server: CASTor Cluster/6.0.0
<html><body><h2>CASTor Error</h2><br>Stream has been modified</body></html>
Retries: 0 ResultCode: ScspRCFailure
```

This example fails because the object was changed by the sample code.

401 (Unauthorized)

401 (Unauthorized) is expected in some SCSPClientExample authentication examples, such as the following:

```
>>>>> Info Domain without Credentials<<<<<< HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest realm="localusers.realm/_administrators",
nonce="2a05cf4e8ff625f5e06bf12aefb0ce86", opaque="e8b019c6b78eb5137b3deb8aabe88cb0",
stale=false, qop="auth", algorithm=MD5
WWW-Authenticate: Basic realm="localusers.realm/_administrators"
Date: Fri, 02 Dec 2011 15:50:29 GMT
Content-Length: 0 Allow: HEAD, GET, PUT, POST, COPY, APPEND, DELETE
Server: CASTor Cluster/6.0.0
Retries: 0 ResultCode: ScspRCFailure
```

This failure is expected because the method requires authentication but no credentials are supplied.

Domains

SCSPClientExample examples that depend on domains being set up fail if the domains are not set up properly. Examples follow:

```
Realm 'allusers.realm' doesn't appear to have been set up correctly: Castor-Authorization header
```

The preceding error indicates the allusers.realm was set up with the wrong domain protection setting.

```
Couldn't find realm 'localusers.realm'. Returning.
```

The preceding error indicates the realm does not exist.

Review the information discussed in “Using SDK Code Examples” in the [SDK Overview](#).

SCSP Proxy

In the event the SCSP Proxy is not set up properly or is unavailable, Java exceptions display, including the following:

```
com.caringo.client.ScspExecutionException: Too many SCSP retries.
```

Java Administrative Override of an Allow Header

This example is not included in the code samples provided with the SDK.

```
// This example shows how to execute a copy command with administrative
// override to clean up Allow headers. Note that you may want to get the
// original headers first to make sure none of the existing ones on the stream
// are lost.

ScspHeaders headers = new ScspHeaders();
headers.addValue("Allow", "PUT, APPEND"); // we're fixing the Allow
// set the credentials
ScspAuthentication auth = new ScspAuthentication("admin", "ourpwdofchoicehere", DEFAULTADMINREALM);
headers.setAuthentication(auth);
ScspQueryArgs args = new ScspQueryArgs();
// make the request administrative args.setValue("admin", "yes");
response = client.copy(uuid, "", args, headers);
```



© 2005–2021 DataCore Software Corporation. All Rights Reserved. DataCore, the DataCore logo, SANsymphony, vFile0, and Swarm are trademarks or registered trademarks of DataCore Software Corporation. All other products, services, and company names mentioned herein may be trademarks of their respective owners.

[Open Source Software Licenses](#) | [EULA](#)

No part of this material may be reproduced, transmitted, or transcribed without the written consent of DataCore Software Corporation. Updates to this material are continuous and are posted as soon as they become available.

datacore.com	Worldwide Headquarters 1901 Cypress Creek Road, Suite 200 Ft. Lauderdale, FL 33309 +1 954-377-6000
DataCore Downloads	Log on to download DataCore software
DataCore Training and Certifications	Access training resources
Support Portal	Log on to enter support tickets and to search the knowledge base, which includes the latest documentation, FAQs, technical notes, product advisories, and troubleshooting.