



Use StorageGRID

StorageGRID

NetApp

March 10, 2025

This PDF was generated from <https://docs.netapp.com/us-en/storagegrid-115/tenant/web-browser-requirements.html> on March 10, 2025. Always check docs.netapp.com for the latest.

Table of Contents

Use StorageGRID	1
Use a tenant account	1
Using the Tenant Manager	1
Managing system access for tenant users	14
Managing S3 tenant accounts	35
Managing S3 platform services	63
Use S3	102
Support for the S3 REST API	103
Configuring tenant accounts and connections	106
How StorageGRID implements the S3 REST API	112
S3 REST API supported operations and limitations	118
StorageGRID S3 REST API operations	168
Bucket and group access policies	189
Configuring security for the REST API	214
Monitoring and auditing operations	217
Benefits of active, idle, and concurrent HTTP connections	220
Use Swift	223
OpenStack Swift API support in StorageGRID	223
Configuring tenant accounts and connections	226
Swift REST API supported operations	231
StorageGRID Swift REST API operations	243
Configuring security for the REST API	248
Monitoring and auditing operations	250

Use StorageGRID

Use a tenant account

Learn how to use a StorageGRID tenant account.

- [Using the Tenant Manager](#)
- [Managing system access for tenant users](#)
- [Managing S3 tenant accounts](#)
- [Managing S3 platform services](#)

Using the Tenant Manager

The Tenant Manager allows you to manage all aspects of a StorageGRID tenant account.

You can use the Tenant Manager to monitor a tenant account's storage usage and to manage users with identity federation or by creating local groups and users. For S3 tenant accounts, you can also manage S3 keys, manage S3 buckets, and configure platform services.

Using a StorageGRID tenant account

A tenant account allows you to use either the Simple Storage Service (S3) REST API or the Swift REST API to store and retrieve objects in a StorageGRID system.

Each tenant account has its own federated or local groups, users, S3 buckets or Swift containers, and objects.

Optionally, tenant accounts can be used to segregate stored objects by different entities. For example, multiple tenant accounts can be used for either of these use cases:

- **Enterprise use case:** If the StorageGRID system is being used within an enterprise, the grid's object storage might be segregated by the different departments in the organization. For example, there might be tenant accounts for the Marketing department, the Customer Support department, the Human Resources department, and so on.



If you use the S3 client protocol, you can also use S3 buckets and bucket policies to segregate objects between the departments in an enterprise. You do not need to create separate tenant accounts. See instructions for implementing S3 client applications.

- **Service provider use case:** If the StorageGRID system is being used by a service provider, the grid's object storage might be segregated by the different entities that lease the storage. For example, there might be tenant accounts for Company A, Company B, Company C, and so on.

Creating tenant accounts

Tenant accounts are created by a StorageGRID grid administrator using the Grid Manager. When creating a tenant account, the grid administrator specifies the following information:

- Display name for the tenant (the tenant's account ID is assigned automatically and cannot be changed).
- Whether the tenant account will use the S3 or Swift.
- For S3 tenant accounts: Whether the tenant account is allowed to use platform services. If the use of

platform services is allowed, the grid must be configured to support their use.

- Optionally, a storage quota for the tenant account—the maximum number of gigabytes, terabytes, or petabytes available for the tenant's objects. A tenant's storage quota represents a logical amount (object size), not a physical amount (size on disk).
- If identity federation is enabled for the StorageGRID system, which federated group has Root Access permission to configure the tenant account.
- If single sign-on (SSO) is not in use for the StorageGRID system, whether the tenant account will use its own identity source or share the grid's identity source, and the initial password for the tenant's local root user.

In addition, grid administrators can enable the S3 Object Lock setting for the StorageGRID system if S3 tenant accounts need to comply with regulatory requirements. When S3 Object Lock is enabled, all S3 tenant accounts can create and manage compliant buckets.

Configuring S3 tenants

After an S3 tenant account is created, you can access the Tenant Manager to perform tasks such as the following:

- Setting up identity federation (unless the identity source is shared with the grid), or creating local groups and users
- Managing S3 access keys
- Creating and managing S3 buckets, including compliant buckets
- Using platform services (if enabled)
- Monitoring storage usage



While you can create and manage S3 buckets with the Tenant Manager, you must have S3 access keys and use the S3 REST API to ingest and manage objects.

Configuring Swift tenants

After a Swift tenant account is created, users with the Root Access permission can access the Tenant Manager to perform tasks such as the following:

- Setting up identity federation (unless the identity source is shared with the grid), and creating local groups and users
- Monitoring storage usage



Swift users must have the Root Access permission to access the Tenant Manager. However, the Root Access permission does not allow users to authenticate into the Swift REST API to create containers and ingest objects. Users must have the Swift Administrator permission to authenticate into the Swift REST API.

Related information

[Administer StorageGRID](#)

[Use S3](#)

[Use Swift](#)

Web browser requirements

You must use a supported web browser.

Web browser	Minimum supported version
Google Chrome	87
Microsoft Edge	87
Mozilla Firefox	84

You should set the browser window to a recommended width.

Browser width	Pixels
Minimum	1024
Optimum	1280

Signing in to the Tenant Manager

You access the Tenant Manager by entering the URL for the tenant into the address bar of a supported web browser.

What you'll need

- You must have your login credentials.
- You must have a URL for accessing the Tenant Manager, as supplied by your grid administrator. The URL will look like one of these examples:

```
https://FQDN_or_Admin_Node_IP/
```

```
https://FQDN_or_Admin_Node_IP:port/
```

```
https://FQDN_or_Admin_Node_IP/?accountId=20-digit-account-id
```

```
https://FQDN_or_Admin_Node_IP:port/?accountId=20-digit-account-id
```

The URL always contains either the fully qualified domain name (FQDN) or the IP address used to access an Admin Node, and could optionally also include a port number, the 20-digit tenant account ID, or both.

- If the URL does not include the tenant's 20-digit account ID, you must have this account ID.

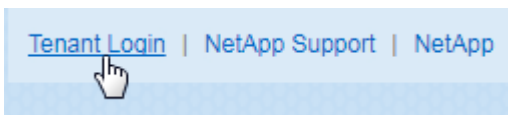
- You must be using a supported web browser.
- Cookies must be enabled in your web browser.
- You must have specific access permissions.

Steps

1. Launch a supported web browser.
2. In the browser's address bar, enter the URL for accessing Tenant Manager.
3. If you are prompted with a security alert, install the certificate using the browser's installation wizard.
4. Sign in to the Tenant Manager.

The sign-in screen that you see depends on the URL you entered and whether your organization is using single sign-on (SSO). You will see one of the following screens:

- The Grid Manager sign-in page. Click the **Tenant Login** link in the upper right.



- The Tenant Manager sign-in page. The **Account ID** field might already be completed, as shown below.

 A screenshot of the 'StorageGRID® Tenant Manager' sign-in page. On the left is the NetApp logo. The main form area has a light blue background and contains the following fields:

- 'Recent' dropdown menu with '-- Optional --' selected.
- 'Account ID' text box containing the value '39105156032765926037'.
- 'Username' text box.
- 'Password' text box.
- 'Sign in' button at the bottom right.

- i. If the tenant's 20-digit account ID is not shown, select the name of the tenant account if it appears in the list of recent accounts, or enter the account ID.
- ii. Enter your username and password.
- iii. Click **Sign in**.

The Tenant Manager Dashboard appears.

- Your organization's SSO page, if SSO is enabled on the grid. For example:

Enter your standard SSO credentials, and click **Sign in**.

- The Tenant Manager SSO sign-in page.

- If the tenant's 20-digit account ID is not shown, select the name of the tenant account if it appears in the list of recent accounts, or enter the account ID.
- Click **Sign in**.
- Sign in with your standard SSO credentials on your organization's SSO sign-in page.

The Tenant Manager Dashboard appears.

- If you received an initial password from someone else, change your password to secure your account. Select **username** > **Change Password**.



If SSO is enabled for the StorageGRID system, you cannot change your password from the Tenant Manager.

Related information

[Administer StorageGRID](#)

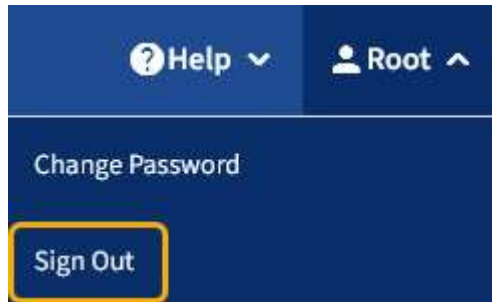
[Web browser requirements](#)

Signing out of the Tenant Manager

When you are done working with the Tenant Manager, you must sign out to ensure that unauthorized users cannot access the StorageGRID system. Closing your browser might not sign you out of the system, based on browser cookie settings.

Steps

1. Locate the username drop-down in the top-right corner of the user interface.



2. Select the username and then select **Sign Out**.

Option	Description
SSO not in use	<p>You are signed out of the Admin Node. The Tenant Manager sign in page is displayed.</p> <p>Note: If you signed into more than one Admin Node, you must sign out of each node.</p>
SSO enabled	<p>You are signed out of all Admin Nodes you were accessing. The StorageGRID Sign in page is displayed. The name of the tenant account you just accessed is listed as the default in the Recent Accounts drop-down, and the tenant's Account ID is shown.</p> <p>Note: If SSO is enabled and you are also signed in to the Grid Manager, you must also sign out of the Grid Manager to sign out of SSO.</p>

Understanding the Tenant Manager Dashboard

The Tenant Manager Dashboard provides an overview of a tenant account's configuration and the amount of space used by objects in the tenant's buckets (S3) or containers (Swift). If the tenant has a quota, the Dashboard shows how much of the quota is used and how much is remaining. If there are any errors related to the tenant account, the errors are shown on the Dashboard.



The Space used values are estimates. These estimates are affected by the timing of ingests, network connectivity, and node status.

When objects have been uploaded, the Dashboard looks like the following example:

Dashboard

16 Buckets
[View buckets](#)

2 Platform services endpoints
[View endpoints](#)

0 Groups
[View groups](#)

1 User
[View users](#)

Storage usage [?](#)

6.5 TB of 7.2 TB used

0.7 TB (10.1%) remaining



Bucket name	Space used	Number of objects
Bucket-15	969.2 GB	913,425
Bucket-04	937.2 GB	576,806
Bucket-13	815.2 GB	957,389
Bucket-06	812.5 GB	193,843
Bucket-10	473.9 GB	583,245
Bucket-03	403.2 GB	981,226
Bucket-07	362.5 GB	420,726
Bucket-05	294.4 GB	785,190
8 other buckets	1.4 TB	3,007,036

Total objects

8,418,886
objects

Tenant details

Name Human Resources
ID 4955 9096 9804 4285 4354

View the instructions for Tenant Manager.

[Go to documentation](#) [↗](#)

Tenant account summary

The top of the Dashboard contains the following information:

- The number of configured buckets or containers, groups, and users
- The number of platform services endpoints, if any have been configured

You can select the links to view the details.

The right side of the Dashboard contains the following information:

- The total number of objects for the tenant.

For an S3 account, if no objects have been ingested and you have the Root Access permission, getting started guidelines appear instead of the total number of objects.

- The tenant account name and ID.
- A link to the StorageGRID documentation.

Storage and quota usage

The Storage usage panel contains the following information:

- The amount of object data for the tenant.



This value indicates the total amount of object data uploaded and does not represent the space used to store copies of those objects and their metadata.

- If a quota is set, the total amount of space available for object data and the amount and percentage of space remaining. The quota limits the amount of object data that can be ingested.












Quota utilization is based on internal estimates and might be exceeded in some cases. For example, StorageGRID checks the quota when a tenant starts uploading objects and rejects new ingests if the tenant has exceeded the quota. However, StorageGRID does not take into account the size of the current upload when determining if the quota has been exceeded. If objects are deleted, a tenant might be temporarily prevented from uploading new objects until the quota utilization is recalculated. Quota utilization calculations can take 10 minutes or longer.

- A bar chart that represents the relative sizes of the largest buckets or containers.

You can place your cursor over any of the chart segments to view the total space consumed by that bucket or container.



- To correspond with the bar chart, a list of the largest buckets or containers, including the total amount of object data and the number of objects for each bucket or container.


Bucket name	Space used	Number of objects
 Bucket-02	944.7 GB	7,575
 Bucket-09	899.6 GB	589,677
 Bucket-15	889.6 GB	623,542
 Bucket-06	846.4 GB	648,619
 Bucket-07	730.8 GB	808,655
 Bucket-04	700.8 GB	420,493
 Bucket-11	663.5 GB	993,729
 Bucket-03	656.9 GB	379,329
 9 other buckets	2.3 TB	5,171,588

If the tenant has more than nine buckets or containers, all other buckets or containers are combined into a single entry at the bottom of the list.


Quota usage alerts

If quota usage alerts have been enabled in the Grid Manager, they will appear in the Tenant Manager when the quota is low or exceeded, as follows:

If 90% or more of a tenant's quota has been used, the **Tenant quota usage high** alert is triggered. For more information, see the alerts reference in the instructions for monitoring and troubleshooting StorageGRID.

 Only 0.6% of the quota is remaining. If the quota is exceeded, you can no longer upload new objects.

If you exceed your quota, you cannot upload new objects.


 The quota has been met. You cannot upload new objects.



To view additional details and manage rules and notifications for alerts, see the instructions for monitoring and troubleshooting StorageGRID.

Endpoint errors

If you have used the Grid Manager to configure one or more endpoints for use with platform services, the Tenant Manager Dashboard displays an alert if any endpoint errors have occurred within the past seven days.

 One or more endpoints have experienced an error and might not be functioning properly. Go to the [Endpoints](#) page to view the error details. The last error occurred 2 hours ago.

To see details about an endpoint error, select Endpoints to display the Endpoints page.

Related information

[Troubleshooting platform services endpoint errors](#)

[Monitor & troubleshoot](#)

Understanding the Tenant Management API

You can perform system management tasks using the Tenant Management REST API instead of the Tenant Manager user interface. For example, you might want to use the API to automate operations or to create multiple entities, such as users, more quickly.

The Tenant Management API uses the Swagger open source API platform. Swagger provides an intuitive user interface that allows developers and non-developers to interact with the API. The Swagger user interface provides complete details and documentation for each API operation.

To access the Swagger documentation for the Tenant Management API:

Steps

1. Sign in to the Tenant Manager.
2. Select **Help > API Documentation** from the Tenant Manager header.

API operations

The Tenant Management API organizes the available API operations into the following sections:

- **account** — Operations on the current tenant account, including getting storage usage information.
- **auth** — Operations to perform user session authentication.

The Tenant Management API supports the Bearer Token Authentication Scheme. For a tenant login, you provide a username, password, and accountId in the JSON body of the authentication request (that is, `POST /api/v3/authorize`). If the user is successfully authenticated, a security token is returned. This token must be provided in the header of subsequent API requests ("Authorization: Bearer token").

See “Protecting against Cross-Site Request Forgery” for information on improving authentication security.



If single sign-on (SSO) is enabled for the StorageGRID system, you must perform different steps to authenticate. See “Authenticating in to the API if single sign-on is enabled” in the instructions for administering StorageGRID.

- **config** — Operations related to the product release and versions of the Tenant Management API. You can list the product release version and the major versions of the API supported by that release.
- **containers** — Operations on S3 buckets or Swift containers, as follows:

Protocol	Permission allows
S3	<ul style="list-style-type: none">• Creating compliant and non-compliant buckets• Modifying legacy compliance settings• Setting the consistency control for operations performed on objects• Creating, updating, and deleting a bucket's CORS configuration• Enabling and disabling last access time updates for objects• Managing the configuration settings for platform services, including CloudMirror replication, notifications, and search integration (metadata-notification)• Deleting empty buckets
Swift	Setting the consistency level used for containers

- **deactivated-features** — Operations to view features that might have been deactivated.
- **endpoints** — Operations to manage an endpoint. Endpoints allow an S3 bucket to use an external service for StorageGRID CloudMirror replication, notifications, or search integration.
- **groups** — Operations to manage local tenant groups and to retrieve federated tenant groups from an external identity source.
- **identity-source** — Operations to configure an external identity source and to manually synchronize federated group and user information.
- **regions** — Operations to determine which regions have been configured for the StorageGRID system.
- **s3** — Operations to manage S3 access keys for tenant users.
- **s3-object-lock** — Operations to determine how global S3 Object Lock (compliance) is configured for the StorageGRID system.

- **users** — Operations to view and manage tenant users.

Operation details

When you expand each API operation, you can see its HTTP action, endpoint URL, a list of any required or optional parameters, an example of the request body (when required), and the possible responses.

groups Operations on groups

GET **/org/groups** Lists Tenant User Groups

Try it out

Parameters

Name	Description
type string (query)	filter by group type
limit integer (query)	maximum number of results
marker string (query)	marker-style pagination offset (value is Group's URN)
includeMarker boolean (query)	if set, the marker element is also returned
order string (query)	pagination order (desc requires marker)

Responses
Response content type application/json

Code	Description
200	<div> <div>Example Value</div> <div>Model</div> <pre>{ "responseTime": "2018-02-01T16:22:31.066Z", "status": "success", "apiVersion": "2.1" }</pre> </div>

Issuing API requests



Any API operations you perform using the API Docs webpage are live operations. Be careful not to create, update, or delete configuration data or other data by mistake.

Steps

1. Click the HTTP action to see the request details.

2. Determine if the request requires additional parameters, such as a group or user ID. Then, obtain these values. You might need to issue a different API request first to get the information you need.
3. Determine if you need to modify the example request body. If so, you can click **Model** to learn the requirements for each field.
4. Click **Try it out**.
5. Provide any required parameters, or modify the request body as required.
6. Click **Execute**.
7. Review the response code to determine if the request was successful.

Related information

[Protecting against Cross-Site Request Forgery \(CSRF\)](#)

[Administer StorageGRID](#)

Tenant Management API versioning

The Tenant Management API uses versioning to support non-disruptive upgrades.

For example, this Request URL specifies version 3 of the API.

```
https://hostname_or_ip_address/api/v3/authorize
```

The major version of the Tenant Management API is bumped when changes are made that are **not compatible** with older versions. The minor version of the Tenant Management API is bumped when changes are made that **are compatible** with older versions. Compatible changes include the addition of new endpoints or new properties. The following example illustrates how the API version is bumped based on the type of changes made.

Type of change to API	Old version	New version
Compatible with older versions	2.1	2.2
Not compatible with older versions	2.1	3.0

When StorageGRID software is installed for the first time, only the most recent version of the Tenant Management API is enabled. However, when StorageGRID is upgraded to a new feature release, you continue to have access to the older API version for at least one StorageGRID feature release.

Outdated requests are marked as deprecated in the following ways:

- The response header is "Deprecated: true"
- The JSON response body includes "deprecated": true

Determining which API versions are supported in the current release

Use the following API request to return a list of the supported API major versions:

```
GET https://{{IP-Address}}/api/versions
{
  "responseTime": "2019-01-10T20:41:00.845Z",
  "status": "success",
  "apiVersion": "3.0",
  "data": [
    2,
    3
  ]
}
```

Specifying an API version for a request

You can specify the API version using a path parameter (`/api/v3`) or a header (`Api-Version: 3`). If you provide both values, the header value overrides the path value.

```
curl https://[IP-Address]/api/v3/grid/accounts

curl -H "Api-Version: 3" https://[IP-Address]/api/grid/accounts
```

Protecting against Cross-Site Request Forgery (CSRF)

You can help protect against Cross-Site Request Forgery (CSRF) attacks against StorageGRID by using CSRF tokens to enhance authentication that uses cookies. The Grid Manager and Tenant Manager automatically enable this security feature; other API clients can choose whether to enable it when they sign in.

An attacker that can trigger a request to a different site (such as with an HTTP form POST) can cause certain requests to be made using the signed-in user's cookies.

StorageGRID helps protect against CSRF attacks by using CSRF tokens. When enabled, the contents of a specific cookie must match the contents of either a specific header or a specific POST body parameter.

To enable the feature, set the `csrfToken` parameter to `true` during authentication. The default is `false`.

```
curl -X POST --header "Content-Type: application/json" --header "Accept:
application/json" -d "{
  \"username\": \"MyUserName\",
  \"password\": \"MyPassword\",
  \"cookie\": true,
  \"csrfToken\": true
}" "https://example.com/api/v3/authorize"
```

When `true`, a `GridCsrfToken` cookie is set with a random value for sign-ins to the Grid Manager, and the `AccountCsrfToken` cookie is set with a random value for sign-ins to the Tenant Manager.

If the cookie is present, all requests that can modify the state of the system (POST, PUT, PATCH, DELETE) must include one of the following:

- The `X-Csrf-Token` header, with the value of the header set to the value of the CSRF token cookie.
- For endpoints that accept a form-encoded body: A `csrfToken` form-encoded request body parameter.

See the online API documentation for additional examples and details.



Requests that have a CSRF token cookie set will also enforce the `"Content-Type: application/json"` header for any request that expects a JSON request body as an additional protection against CSRF attacks.

Managing system access for tenant users

You grant users access to a tenant account by importing groups from a federated identity source and assigning management permissions. You can also create local tenant groups and users, unless single sign-on (SSO) is in effect for the entire StorageGRID system.

- [Using identity federation](#)
- [Managing groups](#)
- [Managing local users](#)

Using identity federation

Using identity federation makes setting up tenant groups and users faster, and it allows tenant users to sign in to the tenant account using familiar credentials.

- [Configuring a federated identity source](#)
- [Forcing synchronization with the identity source](#)
- [Disabling identity federation](#)

Configuring a federated identity source

You can configure identity federation if you want tenant groups and users to be managed in another system such as Active Directory, OpenLDAP, or Oracle Directory Server.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must have specific access permissions.
- You must be using Active Directory, OpenLDAP, or Oracle Directory Server as the identity provider. If you want to use an LDAP v3 service that is not listed, you must contact technical support.
- If you plan to use Transport Layer Security (TLS) for communications with the LDAP server, the identity provider must be using TLS 1.2 or 1.3.

About this task

Whether you can configure an identity federation service for your tenant depends on how your tenant account was set up. Your tenant might share the identity federation service that was configured for the Grid Manager. If you see this message when you access the Identity Federation page, you cannot configure a separate

federated identity source for this tenant.



This tenant account uses the LDAP server that is configured for the Grid Manager. Contact the grid administrator for information or to change this setting.

Steps

1. Select **ACCESS MANAGEMENT > Identity federation**.
2. Select **Enable identity federation**.
3. In the LDAP service type section, select **Active Directory**, **OpenLDAP**, or **Other**.

If you select **OpenLDAP**, configure the OpenLDAP server. See the guidelines for configuring an OpenLDAP server.

Select **Other** to configure values for an LDAP server that uses Oracle Directory Server.

4. If you selected **Other**, complete the fields in the LDAP Attributes section.
 - **User Unique Name:** The name of the attribute that contains the unique identifier of an LDAP user. This attribute is equivalent to `sAMAccountName` for Active Directory and `uid` for OpenLDAP. If you are configuring Oracle Directory Server, enter `uid`.
 - **User UUID:** The name of the attribute that contains the permanent unique identifier of an LDAP user. This attribute is equivalent to `objectGUID` for Active Directory and `entryUUID` for OpenLDAP. If you are configuring Oracle Directory Server, enter `nsuniqueid`. Each user's value for the specified attribute must be a 32-digit hexadecimal number in either 16-byte or string format, where hyphens are ignored.
 - **Group unique name:** The name of the attribute that contains the unique identifier of an LDAP group. This attribute is equivalent to `sAMAccountName` for Active Directory and `cn` for OpenLDAP. If you are configuring Oracle Directory Server, enter `cn`.
 - **Group UUID:** The name of the attribute that contains the permanent unique identifier of an LDAP group. This attribute is equivalent to `objectGUID` for Active Directory and `entryUUID` for OpenLDAP. If you are configuring Oracle Directory Server, enter `nsuniqueid`. Each group's value for the specified attribute must be a 32-digit hexadecimal number in either 16-byte or string format, where hyphens are ignored.
5. In the Configure LDAP server section, enter the required LDAP server and network connection information.
 - **Hostname:** The server hostname or IP address of the LDAP server.
 - **Port:** The port used to connect to the LDAP server. The default port for STARTTLS is 389, and the default port for LDAPS is 636. However, you can use any port as long as your firewall is configured correctly.
 - **Username:** The full path of the distinguished name (DN) for the user that will connect to the LDAP server. For Active Directory, you can also specify the Down-Level Logon Name or the User Principal Name.

The specified user must have permission to list groups and users and to access the following attributes:

- `sAMAccountName` or `uid`
- `objectGUID`, `entryUUID`, or `nsuniqueid`
- `cn`

- `memberOf` or `isMemberOf`
- **Password:** The password associated with the username.
- **Group base DN:** The full path of the distinguished name (DN) for an LDAP subtree you want to search for groups. In the Active Directory example (below), all groups whose Distinguished Name is relative to the base DN (DC=storagegrid,DC=example,DC=com) can be used as federated groups.

The **Group unique name** values must be unique within the **Group base DN** they belong to.

- **User base DN:** The full path of the distinguished name (DN) of an LDAP subtree you want to search for users.

The **User unique name** values must be unique within the **User base DN** they belong to.

6. In the **Transport Layer Security (TLS)** section, select a security setting.

- **Use STARTTLS (recommended):** Use STARTTLS to secure communications with the LDAP server. This is the recommended option.
- **Use LDAPS:** The LDAPS (LDAP over SSL) option uses TLS to establish a connection to the LDAP server. This option is supported for compatibility reasons.
- **Do not use TLS:** The network traffic between the StorageGRID system and the LDAP server will not be secured.

This option is not supported if your Active Directory server enforces LDAP signing. You must use STARTTLS or LDAPS.

7. If you selected STARTTLS or LDAPS, choose the certificate used to secure the connection.

- **Use operating system CA certificate:** Use the default CA certificate installed on the operating system to secure connections.
- **Use custom CA certificate:** Use a custom security certificate.

If you select this setting, copy and paste the custom security certificate into the CA certificate text box.

8. Select **Test connection** to validate your connection settings for the LDAP server.

A confirmation message appears in the upper right corner of the page if the connection is valid.

9. If the connection is valid, select **Save**.

The following screenshot shows example configuration values for an LDAP server that uses Active Directory.

LDAP service type

Select the type of LDAP service you want to configure.

Active Directory

OpenLDAP

Other

Configure LDAP server (All fields are required)

Hostname

my-active-directory.example.com

Port

389

Username

MyDomain\Administrator

Password

••••••••

Group Base DN

DC=storagegrid,DC=example,DC=com

User Base DN

DC=storagegrid,DC=example,DC=com

Related information

[Tenant management permissions](#)

[Guidelines for configuring an OpenLDAP server](#)

Guidelines for configuring an OpenLDAP server

If you want to use an OpenLDAP server for identity federation, you must configure specific settings on the OpenLDAP server.

Memberof and refint overlays

The memberof and refint overlays should be enabled. For more information, see the instructions for reverse

group membership maintenance in the Administrator's Guide for OpenLDAP.

Indexing

You must configure the following OpenLDAP attributes with the specified index keywords:

```
olcDbIndex: objectClass eq
olcDbIndex: uid eq,pres,sub
olcDbIndex: cn eq,pres,sub
olcDbIndex: entryUUID eq
```

In addition, ensure the fields mentioned in the help for Username are indexed for optimal performance.

See the information about reverse group membership maintenance in the Administrator's Guide for OpenLDAP.

Forcing synchronization with the identity source

The StorageGRID system periodically synchronizes federated groups and users from the identity source. You can force synchronization to start if you want to enable or restrict user permissions as quickly as possible.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must have specific access permissions.
- The saved identity source must be enabled.

Steps

1. Select **ACCESS MANAGEMENT > Identity federation**.

The Identity federation page appears. The **Sync server** button is at the top right of the page.



If the saved identity source is not enabled, the **Sync server** button will not be active.

2. Select **Sync server**.

A confirmation message is displayed indicating that synchronization started successfully.

Related information

[Tenant management permissions](#)

Disabling identity federation

If you configured an identity federation service for this tenant, you can temporarily or permanently disable identity federation for tenant groups and users. When identity federation is disabled, there is no communication between the StorageGRID system and the identity source. However, any settings you have configured are retained, allowing you to easily re-enable identity federation in the future.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must have specific access permissions.

About this task

Before you disable identity federation, you should be aware of the following:

- Federated users will be unable to sign in.
- Federated users who are currently signed in will retain access to the tenant account until their session expires, but they will be unable to sign in after their session expires.
- Synchronization between the StorageGRID system and the identity source will not occur.

Steps

1. Select **ACCESS MANAGEMENT > Identity federation**.
2. Deselect the **Enable identity federation** check box.
3. Select **Save**.

Related information

[Tenant management permissions](#)

Managing groups

You assign permissions to user groups to control which tasks tenant users can perform. You can import federated groups from an identity source, such as Active Directory or OpenLDAP, or you can create local groups.



If single sign-on (SSO) is enabled for your StorageGRID system, local users will not be able to sign in to the Tenant Manager, although they can access S3 and Swift resources, based on group permissions.

Tenant management permissions

Before you create a tenant group, consider which permissions you want to assign to that group. Tenant management permissions determine which tasks users can perform using the Tenant Manager or the Tenant Management API. A user can belong to one or more groups. Permissions are cumulative if a user belongs to multiple groups.

To sign in to the Tenant Manager or to use the Tenant Management API, users must belong to a group that has at least one permission. All users who can sign in can perform the following tasks:

- View the dashboard
- Change their own password (for local users)

For all permissions, the group's Access mode setting determines whether users can change settings and perform operations or whether they can only view the related settings and features.



If a user belongs to multiple groups and any group is set to Read-only, the user will have read-only access to all selected settings and features.

You can assign the following permissions to a group. Note that S3 tenants and Swift tenants have different group permissions. Changes might take up to 15 minutes to take effect because of caching.

Permission	Description
Root Access	<p>Provides full access to the Tenant Manager and the Tenant Management API.</p> <p>Note: Swift users must have Root Access permission to sign in to the tenant account.</p>
Administrator	<p>Swift tenants only. Provides full access to the Swift containers and objects for this tenant account</p> <p>Note: Swift users must have the Swift Administrator permission to perform any operations with the Swift REST API.</p>
Manage Your Own S3 Credentials	<p>S3 tenants only. Allows users to create and remove their own S3 access keys. Users who do not have this permission do not see the STORAGE (S3) > My S3 access keys menu option.</p>
Manage All Buckets	<ul style="list-style-type: none">• S3 tenants: Allows users to use the Tenant Manager and the Tenant Management API to create and delete S3 buckets and to manage the settings for all S3 buckets in the tenant account, regardless of S3 bucket or group policies. <p>Users who do not have this permission do not see the Buckets menu option.</p> <ul style="list-style-type: none">• Swift tenants: Allows Swift users to control the consistency level for Swift containers using the Tenant Management API. <p>Note: You can only assign the Manage All Buckets permission to Swift groups from the Tenant Management API. You cannot assign this permission to Swift groups using the Tenant Manager.</p>
Manage Endpoints	<p>S3 tenants only. Allows users to use the Tenant Manager or the Tenant Management API to create or edit endpoints, which are used as the destination for StorageGRID platform services.</p> <p>Users who do not have this permission do not see the Platform services endpoints menu option.</p>

Related information

[Use S3](#)

[Use Swift](#)

Creating groups for an S3 tenant

You can manage permissions for S3 user groups by importing federated groups or creating local groups.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Root Access permission.
- If you plan to import a federated group, you have configured identity federation and the federated group already exists in the configured identity source.

Steps

1. Select **ACCESS MANAGEMENT** > **Groups**.



2. Select **Create group**.
3. Select the **Local group** tab to create a local group, or select the **Federated group** tab to import a group from the previously configured identity source.

If single sign-on (SSO) is enabled for your StorageGRID system, users belonging to local groups will not be able to sign in to the Tenant Manager, although they can use client applications to manage the tenant's resources, based on group permissions.

4. Enter the group's name.
 - **Local group**: Enter both a display name and a unique name. You can edit the display name later.
 - **Federated group**: Enter the unique name. For Active Directory, the unique name is the name associated with the `sAMAccountName` attribute. For OpenLDAP, the unique name is the name associated with the `uid` attribute.
5. Select **Continue**.
6. Select an Access mode. If a user belongs to multiple groups and any group is set to Read-only, the user will have read-only access to all selected settings and features.
 - **Read-write** (default): Users can log into Tenant Manager and manage the tenant configuration.
 - **Read-only**: Users can only view settings and features. They cannot make any changes or perform any operations in the Tenant Manager or Tenant Management API. Local read-only users can change their own passwords.
7. Select the Group permissions for this group.

See the information about tenant management permissions.

8. Select **Continue**.

9. Select a group policy to determine which S3 access permissions the members of this group will have.

- **No S3 Access:** Default. Users in this group do not have access to S3 resources, unless access is granted with a bucket policy. If you select this option, only the root user will have access to S3 resources by default.
- **Read Only Access:** Users in this group have read-only access to S3 resources. For example, users in this group can list objects and read object data, metadata, and tags. When you select this option, the JSON string for a read-only group policy appears in the text box. You cannot edit this string.
- **Full Access:** Users in this group have full access to S3 resources, including buckets. When you select this option, the JSON string for a full-access group policy appears in the text box. You cannot edit this string.
- **Custom:** Users in the group are granted the permissions you specify in the text box. See the instructions for implementing an S3 client application for detailed information about group policies, including language syntax and examples.

10. If you selected **Custom**, enter the group policy. Each group policy has a size limit of 5,120 bytes. You must enter a valid JSON formatted string.

In this example, members of the group are only permitted to list and access a folder matching their username (key prefix) in the specified bucket. Note that access permissions from other group policies and the bucket policy should be considered when determining the privacy of these folders.

The screenshot shows the AWS IAM console interface for creating a group policy. On the left, there are four radio button options: "No S3 Access", "Read Only Access", "Full Access", and "Custom". The "Custom" option is selected, and below it, a note says "(Must be a valid JSON formatted string.)". To the right of the options is a large text area containing a JSON policy string. The JSON string defines two statements: one for listing buckets and another for accessing objects, both restricted to a specific folder based on the user's username.

```
{
  "Statement": [
    {
      "Sid": "AllowListBucketOfASpecificUserPrefix",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::department-bucket",
      "Condition": {
        "StringLike": {
          "s3:prefix": "${aws:username}/*"
        }
      }
    },
    {
      "Sid": "AllowUserSpecificActionsOnlyInTheSpecificFolder",
      "Effect": "Allow",
      "Action": "s3:*Object",
      "Resource": "arn:aws:s3:::department-bucket/${aws:username}/*"
    }
  ]
}
```

11. Select the button that appears, depending on whether you are creating a federated group or a local group:

- Federated group: **Create group**

- Local group: **Continue**

If you are creating a local group, step 4 (Add users) appears after you select **Continue**. This step does not appear for federated groups.

12. Select the check box for each user you want to add to the group, then select **Create group**.

Optionally, you can save the group without adding users. You can add users to the group later, or select the group when you add new users.

13. Select **Finish**.

The group you created appears in the list of groups. Changes might take up to 15 minutes to take effect because of caching.

Related information

[Tenant management permissions](#)

[Use S3](#)

Creating groups for a Swift tenant

You can manage access permissions for a Swift tenant account by importing federated groups or creating local groups. At least one group must have the Swift Administrator permission, which is required to manage the containers and objects for a Swift tenant account.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Root Access permission.
- If you plan to import a federated group, you have configured identity federation and the federated group already exists in the configured identity source.

Steps

1. Select **ACCESS MANAGEMENT > Groups**.

Groups

Create and manage local and federated groups. Set group permissions to control access to specific pages and features.

2 groups Create group

Actions

<input type="checkbox"/>	Name	ID	Type	Access mode
<input type="checkbox"/>	Applications	22cc2e27-88ee-4461-a8c6-30b550beec0	Local	Read-write
<input type="checkbox"/>	Managers	8b15b131-1d21-4539-93ad-f2298347c4d8	Local	Read-write

Previous **1** Next

2. Select **Create group**.
3. Select the **Local group** tab to create a local group, or select the **Federated group** tab to import a group from the previously configured identity source.

If single sign-on (SSO) is enabled for your StorageGRID system, users belonging to local groups will not be able to sign in to the Tenant Manager, although they can use client applications to manage the tenant's resources, based on group permissions.

4. Enter the group's name.
 - **Local group**: Enter both a display name and a unique name. You can edit the display name later.
 - **Federated group**: Enter the unique name. For Active Directory, the unique name is the name associated with the `sAMAccountName` attribute. For OpenLDAP, the unique name is the name associated with the `uid` attribute.
5. Select **Continue**.
6. Select an Access mode. If a user belongs to multiple groups and any group is set to Read-only, the user will have read-only access to all selected settings and features.
 - **Read-write** (default): Users can log into Tenant Manager and manage the tenant configuration.
 - **Read-only**: Users can only view settings and features. They cannot make any changes or perform any operations in the Tenant Manager or Tenant Management API. Local read-only users can change their own passwords.
7. Set the Group permission.
 - Select the **Root Access** check box if users need to sign in to the Tenant Manager or Tenant Management API. (Default)
 - Unselect the **Root Access** check box if users do not need access to the Tenant Manager or Tenant Management API. For example, unselect the check box for applications that do not need to access the tenant. Then, assign the **Swift Administrator** permission to allow these users to manage containers and objects.
8. Select **Continue**.

9. Select the **Swift administrator** check box if the user needs to be able to use the Swift REST API.

Swift users must have the Root Access permission to access the Tenant Manager. However, the Root Access permission does not allow users to authenticate into the Swift REST API to create containers and ingest objects. Users must have the Swift Administrator permission to authenticate into the Swift REST API.

10. Select the button that appears, depending on whether you are creating a federated group or a local group:
 - Federated group: **Create group**
 - Local group: **Continue**

If you are creating a local group, step 4 (Add users) appears after you select **Continue**. This step does not appear for federated groups.

11. Select the check box for each user you want to add to the group, then select **Create group**.

Optionally, you can save the group without adding users. You can add users to the group later, or select the group when you create new users.

12. Select **Finish**.

The group you created appears in the list of groups. Changes might take up to 15 minutes to take effect because of caching.

Related information

[Tenant management permissions](#)

[Use Swift](#)

Viewing and editing group details

When you view the details for a group, you can change the group's display name, permissions, policies, and the users that belong to the group.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Root Access permission.

Steps

1. Select **ACCESS MANAGEMENT > Groups**.
2. Select the name of the group whose details you want to view or edit.

Alternatively, you can select **Actions > View group details**.

The group details page appears. The following example shows the S3 group details page.

Overview

Display name:	Applications 
Unique name:	group/Applications
Type:	Local
Access mode:	Read-write
Permissions:	Root Access
S3 Policy:	None
Number of users in this group:	0

Group permissions

S3 group policy

Users

Manage group permissions

Select an access mode for this group and select one or more permissions.

Access mode

Select whether users can change settings and perform operations or whether they can only view settings and features.

☒ Read-write ☐ Read-only

Group permissions

Select the tenant account permissions you want to assign to this group.

☒ **Root Access**

Allows users to access all Tenant Manager features. Root Access permission supersedes all other permissions.

☒ **Manage All Buckets**

Allows users to change settings of all S3 buckets (or Swift containers) in this account.

☒ **Manage Endpoints**

Allows users to configure endpoints for platform services.

☒ **Manage Your Own S3 Credentials**

Allows users to create and delete their own S3 access keys.

Save changes

3. Make changes to the group settings as needed.



To ensure your changes are saved, select **Save changes** after you make changes in each section. When your changes are saved, a confirmation message appears in the upper right corner of the page.

- a. Optionally, select the display name or edit icon  to update the display name.

You cannot change a group's unique name. You cannot edit the display name for a federated group.

- b. Optionally, update the permissions.

- c. For group policy, make the appropriate changes for your S3 or Swift tenant.

- If you are editing a group for an S3 tenant, optionally select a different S3 group policy. If you select a custom S3 policy, update the JSON string as required.
- If you are editing a group for a Swift tenant, optionally select or unselect the **Swift Administrator** check box.

For more information about the Swift Administrator permission, see the instructions for creating groups for a Swift tenant.

- d. Optionally, add or remove users.

4. Confirm that you have selected **Save changes** for each section you changed.

Changes might take up to 15 minutes to take effect because of caching.

Related information

[Creating groups for an S3 tenant](#)

[Creating groups for a Swift tenant](#)

Adding users to a local group

You can add users to a local group as needed.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Root Access permission.

Steps

1. Select **ACCESS MANAGEMENT > Groups**.
2. Select the name of the local group you want to add users to.

Alternatively, you can select **Actions > View group details**.

The group details page appears.

Overview

Display name:	Applications 
Unique name:	group/Applications
Type:	Local
Access mode:	Read-write
Permissions:	Root Access
S3 Policy:	None
Number of users in this group:	0

Group permissions

S3 group policy

Users

Manage group permissions

Select an access mode for this group and select one or more permissions.

Access mode

Select whether users can change settings and perform operations or whether they can only view settings and features.

☒ Read-write ☐ Read-only

Group permissions

Select the tenant account permissions you want to assign to this group.

☒ **Root Access**

Allows users to access all Tenant Manager features. Root Access permission supersedes all other permissions.

☒ **Manage All Buckets**

Allows users to change settings of all S3 buckets (or Swift containers) in this account.

☒ **Manage Endpoints**

Allows users to configure endpoints for platform services.

☒ **Manage Your Own S3 Credentials**

Allows users to create and delete their own S3 access keys.

Save changes

3. Select **Manage Users**, and then select **Add users**.

Manage users

You can add users to this group or remove users from this group.

Add users **Remove Users** Search Groups... Displaying 1 results

Username	Full Name	Denied
User_02	User_02_Managers	

4. Select the users you want to add to the group, and then select **Add users**.

Add users ×

Select local users to add to the group **Applications**.

Search Groups... Displaying 1 results

<input checked="" type="checkbox"/>	Username	Full Name	Denied
<input checked="" type="checkbox"/>	User_01	User_01_Applications	

Cancel **Add users**

A confirmation message appears in the upper right corner of the page. Changes might take up to 15 minutes to take effect because of caching.

Editing a group name

You can edit the display name for a group. You cannot edit the unique name for a group.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Root Access permission.

Steps

1. Select **ACCESS MANAGEMENT > Groups**.
2. Select the check box for the group whose display name you want to edit.
3. Select **Actions > Edit group name**.

The Edit group name dialog box appears.

Edit group name ✕

Specify a new name for the group **Applications**.

Must contain at least 1 and no more than 32 characters

Applications

Cancel Save changes

4. If you are editing a local group, update the display name as needed.

You cannot change a group's unique name. You cannot edit the display name for a federated group.

5. Select **Save changes**.

A confirmation message appears in the upper right corner of the page. Changes might take up to 15 minutes to take effect because of caching.

Related information

[Tenant management permissions](#)

Duplicating a group

You can create new groups more quickly by duplicating an existing group.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Root Access permission.

Steps

1. Select **ACCESS MANAGEMENT > Groups**.
2. Select the check box for the group you want to duplicate.
3. Select **Duplicate group**. For additional details on creating a group, see the instructions for creating groups for an S3 tenant or for a Swift tenant.
4. Select the **Local group** tab to create a local group, or select the **Federated group** tab to import a group from the previously configured identity source.

If single sign-on (SSO) is enabled for your StorageGRID system, users belonging to local groups will not be able to sign in to the Tenant Manager, although they can use client applications to manage the tenant's resources, based on group permissions.

5. Enter the group's name.
 - **Local group**: Enter both a display name and a unique name. You can edit the display name later.
 - **Federated group**: Enter the unique name. For Active Directory, the unique name is the name

associated with the `sAMAccountName` attribute. For OpenLDAP, the unique name is the name associated with the `uid` attribute.

6. Select **Continue**.
7. As needed, modify the permissions for this group.
8. Select **Continue**.
9. As needed, if you are duplicating a group for an S3 tenant, optionally select a different policy from the **Add S3 policy** radio buttons. If you selected a custom policy, update the JSON string as required.
10. Select **Create group**.

Related information

[Creating groups for an S3 tenant](#)

[Creating groups for a Swift tenant](#)

[Tenant management permissions](#)

Deleting a group

You can delete a group from the system. Any users who belong only to that group will no longer be able to sign in to the Tenant Manager or use the tenant account.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Root Access permission.

Steps

1. Select **ACCESS MANAGEMENT > Groups**.

Groups

Create and manage local and federated groups. Set group permissions to control access to specific pages and features.

2 groups Create group

Actions

<input type="checkbox"/>	Name	ID	Type	Access mode
<input type="checkbox"/>	Applications	22cc2e27-88ee-4461-a8c6-30b550beeec0	Local	Read-write
<input type="checkbox"/>	Managers	8b15b131-1d21-4539-93ad-f2298347c4d8	Local	Read-write

← Previous **1** Next →

2. Select the check boxes for the groups you want to delete.

3. Select **Actions > Delete group**.

A confirmation message appears.

4. Select **Delete group** to confirm you want to delete the groups indicated in the confirmation message.

A confirmation message appears in the upper right corner of the page. Changes might take up to 15 minutes to take effect because of caching.

Related information

[Tenant management permissions](#)

Managing local users

You can create local users and assign them to local groups to determine which features these users can access. The Tenant Manager includes one predefined local user, named “root.” Although you can add and remove local users, you cannot remove the root user.

What you’ll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a read-write user group that has the Root Access permission.



If single sign-on (SSO) is enabled for your StorageGRID system, local users will not be able to sign in to the Tenant Manager or the Tenant Management API, although they can use S3 or Swift client applications to access the tenant’s resources, based on group permissions.

Accessing the Users page

Select **ACCESS MANAGEMENT > Users**.

Users

View local and federated users. Edit properties and group membership of local users.

3 users

Create user

Actions

<input type="checkbox"/>	Username	Full Name	Denied	Type
<input type="checkbox"/>	root	Root		Local
<input type="checkbox"/>	User_01	User_01		Local
<input type="checkbox"/>	User_02	User_02		Local

Creating local users

You can create local users and assign them to one or more local groups to control their access permissions.

S3 users who do not belong to any groups do not have management permissions or S3 group policies applied to them. These users might have S3 bucket access granted through a bucket policy.

Swift users who do not belong to any groups do not have management permissions or Swift container access.

Steps

1. Select **Create user**.
2. Complete the following fields.
 - **Full name**: The full name for this user, for example, the first name and last name of a person or the name of an application.
 - **Username**: The name this user will use to sign in. Usernames must be unique and cannot be changed.
 - **Password**: A password, which is used when the user signs in.
 - **Confirm password**: Type the same password you typed in the Password field.
 - **Deny access**: If you select **Yes**, this user cannot sign in to the tenant account, even though the user might still belong to one or more groups.

As an example, you can use this feature to temporarily suspend a user's ability to sign in.

3. Select **Continue**.
4. Assign the user to one or more local groups.

Users who do not belong to any groups will have no management permissions. Permissions are cumulative. Users will have all permissions for all groups they belong to.

5. Select **Create user**.

Changes might take up to 15 minutes to take effect because of caching.

Editing user details


When you edit the details for a user, you can change the user's full name and password, add the user to different groups, and prevent the user from accessing the tenant.

Steps

1. In the Users list, select the name of the user whose details you want to view or edit.

Alternatively, you can select the check box for the user, and then select **Actions > View user details**.

2. Make changes to the user settings as needed.

- a. Change the user's full name as needed by selecting the full name or the edit icon  in the Overview section.

You cannot change the username.

- b. On the **Password** tab, change the user's password as needed.
- c. On the **Access** tab, allow the user to sign in (select **No**), or prevent the user from signing in (select **Yes**) as needed.
- d. On the **Groups** tab, add the user to groups or remove the user from groups as needed.
- e. As necessary for each section, select **Save changes**.

Changes might take up to 15 minutes to take effect because of caching.

Duplicating local users

You can duplicate a local user to create a new user more quickly.

Steps

1. In the Users list, select the user you want to duplicate.
2. Select **Duplicate user**.
3. Modify the following fields for the new user.
 - **Full name**: The full name for this user, for example, the first name and last name of a person or the name of an application.
 - **Username**: The name this user will use to sign in. Usernames must be unique and cannot be changed.
 - **Password**: A password, which is used when the user signs in.
 - **Confirm password**: Type the same password you typed in the Password field.
 - **Deny access**: If you select **Yes**, this user cannot sign in to the tenant account, even though the user might still belong to one or more groups.

As an example, you can use this feature to temporarily suspend a user's ability to sign in.

4. Select **Continue**.
5. Select one or more local groups.

Users who do not belong to any groups will have no management permissions. Permissions are cumulative. Users will have all permissions for all groups they belong to.

6. Select **Create user**.

Changes might take up to 15 minutes to take effect because of caching.

Deleting local users

You can permanently delete local users who no longer need to access the StorageGRID tenant account.

Using the Tenant Manager, you can delete local users, but not federated users. You must use the federated identity source to delete federated users.

Steps

1. In the Users list, select the check box for the local user you want to delete.
2. Select **Actions > Delete user**.
3. In the confirmation dialog box, select **Delete user** to confirm you want to delete the user from the system.

Changes might take up to 15 minutes to take effect because of caching.

Related information

[Tenant management permissions](#)

Managing S3 tenant accounts

You can use the Tenant Manager to manage S3 access keys and to create and manage S3 buckets.

- [Managing S3 access keys](#)
- [Managing S3 buckets](#)

Managing S3 access keys

Each user of an S3 tenant account must have an access key to store and retrieve objects in the StorageGRID system. An access key consists of an access key ID and a secret access key.

About this task

S3 access keys can be managed as follows:

- Users who have the **Manage Your Own S3 Credentials** permission can create or remove their own S3 access keys.
- Users who have the **Root Access** permission can manage the access keys for the S3 root account and all other users. Root access keys provide full access to all buckets and objects for the tenant unless explicitly disabled by a bucket policy.

StorageGRID supports Signature Version 2 and Signature Version 4 authentication. Cross-account access is not permitted unless explicitly enabled by a bucket policy.

Creating your own S3 access keys

If you are using an S3 tenant and you have the appropriate permission, you can create your own S3 access keys. You must have an access key to access your buckets and objects in the S3 tenant account.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must have the Manage Your Own S3 Credentials permission.

About this task

You can create one or more S3 access keys that allow you to create and manage buckets for your tenant account. After you create a new access key, update the application with your new access key ID and secret access key. For security, do not create more keys than you need, and delete the keys you are not using. If you have only one key and it is about to expire, create a new key before the old one expires, and then delete the old one.

Each key can have a specific expiration time or no expiration. Follow these guidelines for expiration time:

- Set an expiration time for your keys to limit your access to a certain time period. Setting a short expiration time can help reduce your risk if your access key ID and secret access key are accidentally exposed. Expired keys are removed automatically.
- If the security risk in your environment is low and you do not need to periodically create new keys, you do not have to set an expiration time for your keys. If you decide later to create new keys, delete the old keys manually.



The S3 buckets and objects belonging to your account can be accessed using the access key ID and secret access key displayed for your account in the Tenant Manager. For this reason, protect access keys as you would a password. Rotate access keys on a regular basis, remove any unused keys from your account, and never share them with other users.

Steps

1. Select **STORAGE (S3) > My access keys**.

The My access keys page appears and lists any existing access keys.

2. Select **Create key**.
3. Do one of the following:
 - Select **Do not set an expiration time** to create a key that will not expire. (Default)
 - Select **Set an expiration time**, and set the expiration date and time.

Create access key

1 Choose expiration time

2 Download access key

Choose expiration time

☐ Do not set an expiration time

☒ Set an expiration time

This access key will never expire.

MM/DD/YYYY

HH

:

MM

AM

Cancel

Create access key

4. Select **Create access key**.

The Download access key dialog box appears, listing your access key ID and secret access key.

5. Copy the access key ID and the secret access key to a safe location, or select **Download .csv** to save a spreadsheet file containing the access key ID and secret access key.



Do not close this dialog box until you have copied or downloaded this information.

Create access key

✓ Choose expiration time

2 Download access key

Download access key

To save the keys for future reference, select **Download .csv**, or copy and paste the values to another location.

i

You will not be able to view the Access key ID or Secret access key after you close this dialog.

Access key ID

SHTBQKDRVHQ34YKKUAWX

Secret access key

UGu9+XeACtnOWQYFdbzmngmVXXDvCkSOzT1Osz9K

Download .csv

Finish

6. Select **Finish**.

The new key is listed on the My access keys page. Changes might take up to 15 minutes to take effect because of caching.

Related information

[Tenant management permissions](#)

Viewing your S3 access keys

If you are using an S3 tenant and you have the appropriate permission, you can view a list of your S3 access keys. You can sort the list by expiration time, so you can determine which keys will expire soon. As needed, you can create new keys or delete keys that you are no longer using.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must have the Manage Your Own S3 Credentials permission.

The S3 buckets and objects belonging to your account can be accessed using the access key ID and secret access key displayed for your account in the Tenant Manager. For this reason, protect access keys as you would a password. Rotate access keys on a regular basis, remove any unused keys from your account, and never share them with other users.

Steps

38

1. Select **STORAGE (S3) > My access keys**.

The My access keys page appears and lists any existing access keys.

My access keys

Manage your personal S3 access keys. If a key will expire soon, you can create a new key and delete the one it is replacing.

4 keys Create key

Delete key

<input type="checkbox"/>	Access key ID	Expiration time
<input type="checkbox"/>	*****OTLS	2020-11-23 12:00:00 MST
<input type="checkbox"/>	*****0M45	2020-12-01 19:00:00 MST
<input type="checkbox"/>	*****69QJ	None
<input type="checkbox"/>	*****3R8P	None

2. Sort the keys by **Expiration time** or **Access key ID**.
3. As needed, create new keys and manually delete keys that you are no longer using.

If you create new keys before the existing keys expire, you can begin using the new keys without temporarily losing access to the objects in the account.

Expired keys are removed automatically.

Related information

[Creating your own S3 access keys](#)

[Deleting your own S3 access keys](#)

Deleting your own S3 access keys

If you are using an S3 tenant and you have the appropriate permission, you can delete your own S3 access keys. After an access key is deleted, it can no longer be used to access the objects and buckets in the tenant account.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must have the Manage Your Own S3 Credentials permission.



The S3 buckets and objects belonging to your account can be accessed using the access key ID and secret access key displayed for your account in the Tenant Manager. For this reason, protect access keys as you would a password. Rotate access keys on a regular basis, remove any unused keys from your account, and never share them with other users.

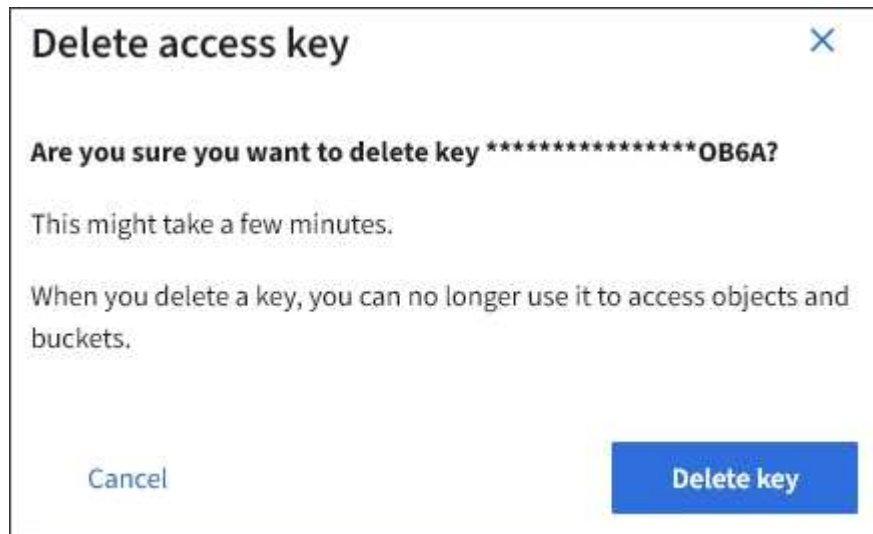
Steps

1. Select **STORAGE (S3) > My access keys**.

The My access keys page appears and lists any existing access keys.

2. Select the check box for each access key you want to remove.
3. Select **Delete key**.

A confirmation dialog box appears.



4. Select **Delete key**.

A confirmation message appears in the upper right corner of the page. Changes might take up to 15 minutes to take effect because of caching.

Related information

[Tenant management permissions](#)

Creating another user's S3 access keys

If you are using an S3 tenant and you have the appropriate permission, you can create S3 access keys for other users, such as applications that need access to buckets and objects.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.

- You must have the Root Access permission.

About this task

You can create one or more S3 access keys for other users so they can create and manage buckets for their tenant account. After you create a new access key, update the application with the new access key ID and secret access key. For security, do not create more keys than the user needs, and delete the keys that are not being used. If you have only one key and it is about to expire, create a new key before the old one expires, and then delete the old one.

Each key can have a specific expiration time or no expiration. Follow these guidelines for expiration time:

- Set an expiration time for the keys to limit the user's access to a certain time period. Setting a short expiration time can help reduce risk if the access key ID and secret access key are accidentally exposed. Expired keys are removed automatically.
- If the security risk in your environment is low and you do not need to periodically create new keys, you do not have to set an expiration time for the keys. If you decide later to create new keys, delete the old keys manually.



The S3 buckets and objects belonging to a user can be accessed using the access key ID and secret access key displayed for that user in the Tenant Manager. For this reason, protect access keys as you would a password. Rotate access keys on a regular basis, remove any unused keys from the account, and never share them with other users.

Steps

1. Select **ACCESS MANAGEMENT > Users**.
2. Select the user whose S3 access keys you want to manage.

The user detail page appears.

3. Select **Access keys**, then select **Create key**.
4. Do one of the following:
 - Select **Do not set an expiration time** to create a key that does not expire. (Default)
 - Select **Set an expiration time**, and set the expiration date and time.

Create access key

1 Choose expiration time

2 Download access key

Choose expiration time

☐ Do not set an expiration time

☒ Set an expiration time

This access key will never expire.

MM/DD/YYYY

HH

:

MM

AM


Cancel

Create access key

5. Select **Create access key**.

The Download access key dialog box appears, listing the access key ID and secret access key.

6. Copy the access key ID and the secret access key to a safe location, or select **Download .csv** to save a spreadsheet file containing the access key ID and secret access key.

 Do not close this dialog box until you have copied or downloaded this information.

Create access key

✓ Choose expiration time 2 Download access key

Download access key

To save the keys for future reference, select **Download .csv**, or copy and paste the values to another location.

i You will not be able to view the Access key ID or Secret access key after you close this dialog.

Access key ID

SHTBQKDRVHQ34YKKUAWX



Secret access key

UGu9+XeACtnOWQYFdbzmngmgVXXDvCkSOzT1Osz9K

[Download .csv](#)

Finish

7. Select **Finish**.

The new key is listed on the Access keys tab of the user details page. Changes might take up to 15 minutes to take effect because of caching.

Related information

Tenant management permissions

Viewing another user's S3 access keys

If you are using an S3 tenant and you have appropriate permissions, you can view another user's S3 access keys. You can sort the list by expiration time so you can determine which keys will expire soon. As needed, you can create new keys and delete keys that are no longer in use.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must have the Root Access permission.



The S3 buckets and objects belonging to a user can be accessed using the access key ID and secret access key displayed for that user in the Tenant Manager. For this reason, protect access keys as you would a password. Rotate access keys on a regular basis, remove any unused keys from the account, and never share them with other users.

Steps

1. Select **ACCESS MANAGEMENT > Users**.

The Users page appears and lists the existing users.

2. Select the user whose S3 access keys you want to view.

The User details page appears.

3. Select **Access keys**.

Manage access keys
Add or delete access keys for this user.

Create key Actions ▾

Displaying 4 results

<input type="checkbox"/>	Access key ID ▴ ▾	Expiration time ▴ ▾
<input type="checkbox"/>	*****WX5J	2020-11-21 12:00:00 MST
<input type="checkbox"/>	*****6OHM	2020-11-23 13:00:00 MST
<input type="checkbox"/>	*****J505	None
<input type="checkbox"/>	*****4MTF	None

4. Sort the keys by **Expiration time** or **Access key ID**.
5. As needed, create new keys and manually delete keys that the are no longer in use.

If you create new keys before the existing keys expire, the user can begin using the new keys without temporarily losing access to the objects in the account.

Expired keys are removed automatically.

Related information

[Creating another user's S3 access keys](#)

[Deleting another user's S3 access keys](#)

Deleting another user's S3 access keys

If you are using an S3 tenant and you have appropriate permissions, you can delete another user's S3 access keys. After an access key is deleted, it can no longer be used to access the objects and buckets in the tenant account.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must have the Root Access permission.



The S3 buckets and objects belonging to a user can be accessed using the access key ID and secret access key displayed for that user in the Tenant Manager. For this reason, protect access keys as you would a password. Rotate access keys on a regular basis, remove any unused keys from the account, and never share them with other users.

Steps

1. Select **ACCESS MANAGEMENT > Users**.

The Users page appears and lists the existing users.

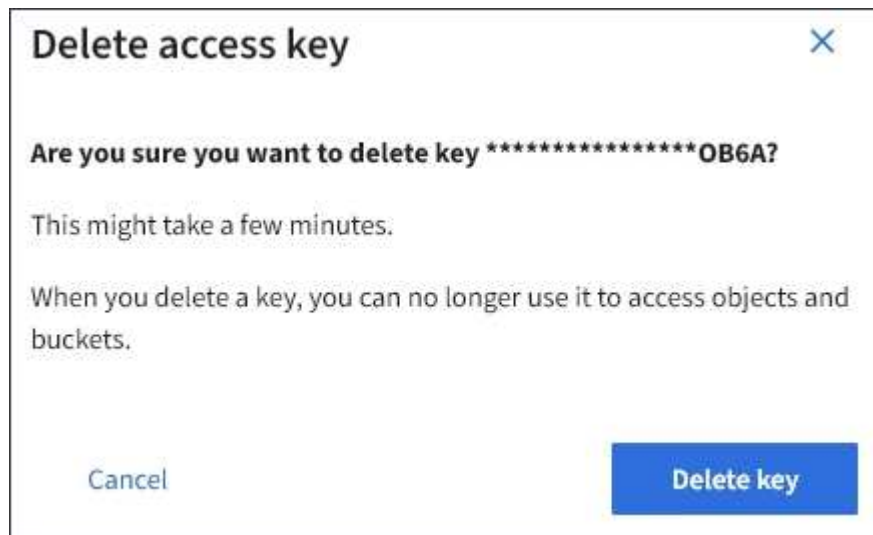
2. Select the user whose S3 access keys you want to manage.

The User details page appears.

3. Select **Access keys**, and then select the check box for each access key you want to delete.

4. Select **Actions > Delete selected key**.

A confirmation dialog box appears.



5. Select **Delete key**.

A confirmation message appears in the upper right corner of the page. Changes might take up to 15 minutes to take effect because of caching.

Related information

[Tenant management permissions](#)

Managing S3 buckets

If you are using an S3 tenant with the appropriate permissions, you can create, view, and delete S3 buckets, update consistency level settings, configure Cross-Origin Resource Sharing (CORS), enable and disable last access time update settings, and manage S3 platform services.

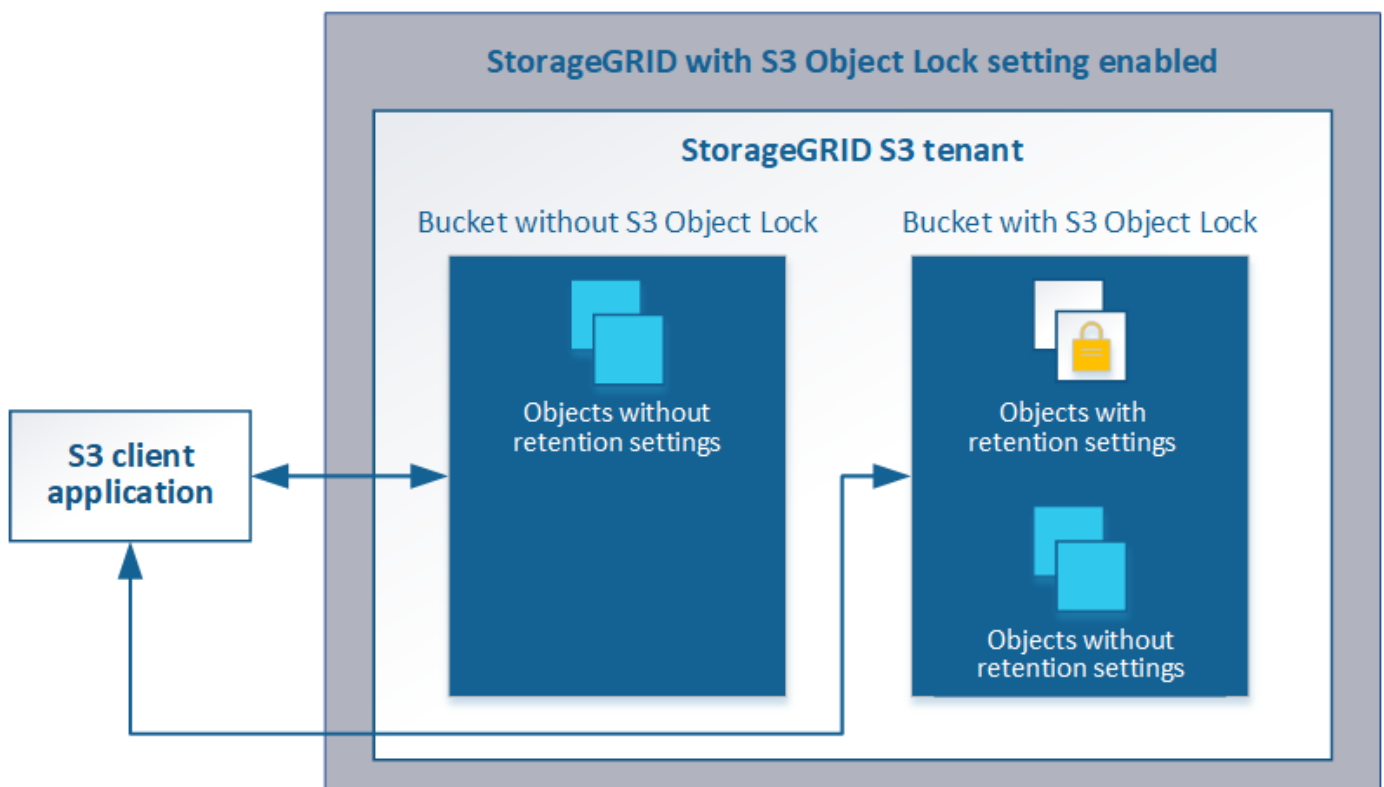
Using S3 Object Lock

You can use the S3 Object Lock feature in StorageGRID if your objects must comply with regulatory requirements for retention.

What is S3 Object Lock?

The StorageGRID S3 Object Lock feature is an object-protection solution that is equivalent to S3 Object Lock in Amazon Simple Storage Service (Amazon S3).

As shown in the figure, when the global S3 Object Lock setting is enabled for a StorageGRID system, an S3 tenant account can create buckets with or without S3 Object Lock enabled. If a bucket has S3 Object Lock enabled, S3 client applications can optionally specify retention settings for any object version in that bucket. An object version must have retention settings specified to be protected by S3 Object Lock.



The StorageGRID S3 Object Lock feature provides a single retention mode that is equivalent to the Amazon S3 compliance mode. By default, a protected object version cannot be overwritten or deleted by any user. The StorageGRID S3 Object Lock feature does not support a governance mode, and it does not allow users with special permissions to bypass retention settings or to delete protected objects.

If a bucket has S3 Object Lock enabled, the S3 client application can optionally specify either or both of the following object-level retention settings when creating or updating an object:

- **Retain-until-date:** If an object version's retain-until-date is in the future, the object can be retrieved, but it cannot be modified or deleted. As required, an object's retain-until-date can be increased, but this date cannot be decreased.
- **Legal hold:** Applying a legal hold to an object version immediately locks that object. For example, you might need to put a legal hold on an object that is related to an investigation or legal dispute. A legal hold has no expiration date, but remains in place until it is explicitly removed. Legal holds are independent of the retain-until-date.

For details on these settings, go to “using S3 object lock” in [S3 REST API supported operations and limitations](#).

Managing legacy Compliant buckets

The S3 Object Lock feature replaces the Compliance feature that was available in previous StorageGRID versions. If you created compliant buckets using a previous version of StorageGRID, you can continue to manage the settings of these buckets; however, you can no longer create new compliant buckets. For instructions, see the NetApp Knowledge Base article.

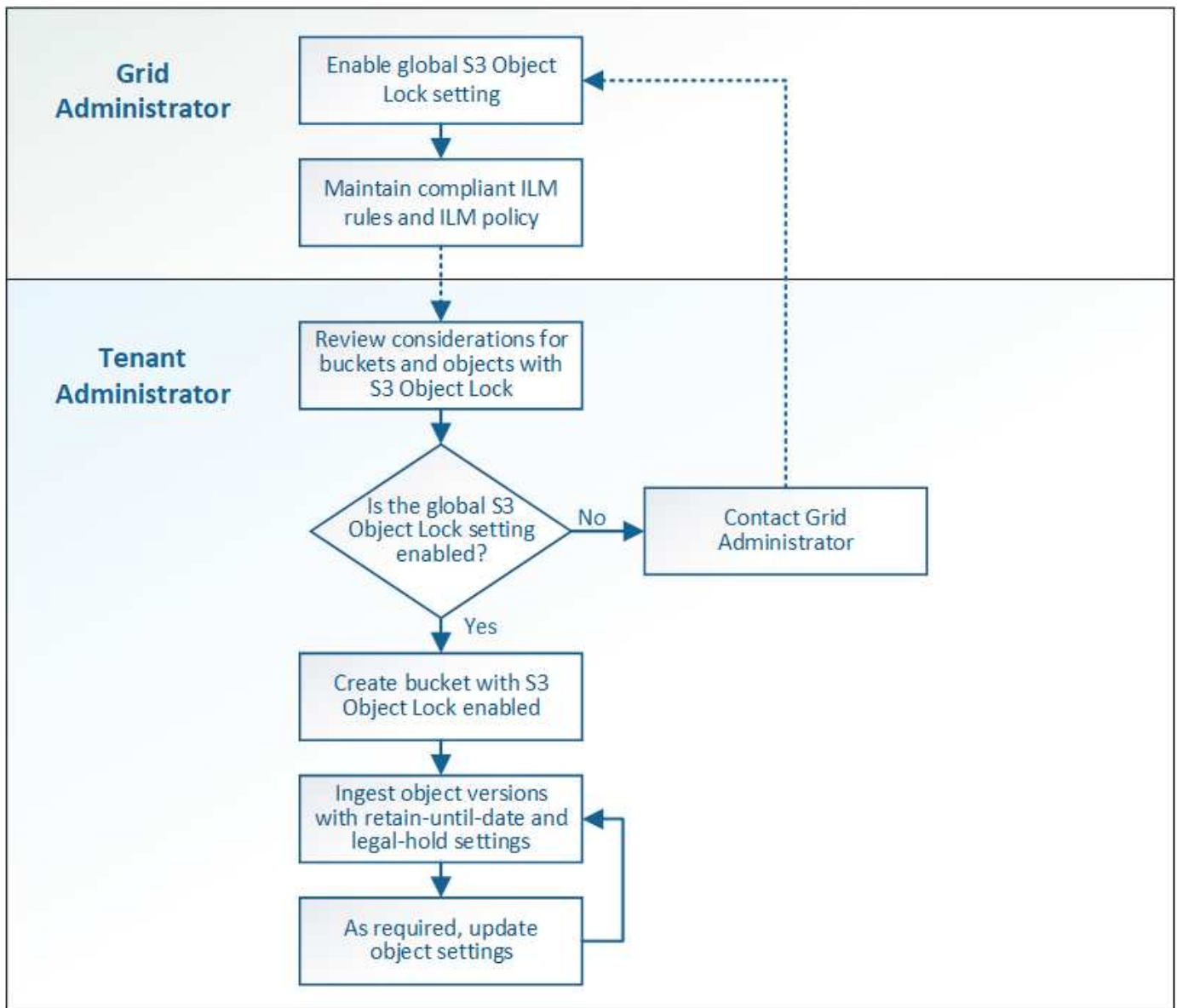
[NetApp Knowledge Base: How to manage legacy Compliant buckets in StorageGRID 11.5](#)

S3 Object Lock workflow

The workflow diagram shows the high-level steps for using the S3 Object Lock feature in StorageGRID.

Before you can create buckets with S3 Object Lock enabled, the grid administrator must enable the global S3 Object Lock setting for the entire StorageGRID system. The grid administrator must also ensure that the information lifecycle management (ILM) policy is “compliant”; it must meet the requirements of buckets with S3 Object Lock enabled. For details, contact your grid administrator or see the instructions for managing objects with information lifecycle management.

After the global S3 Object Lock setting has been enabled, you can create buckets with S3 Object Lock enabled. You can then use the S3 client application to optionally specify retention settings for each object version.



Related information

[Manage objects with ILM](#)

Requirements for S3 Object Lock

Before enabling S3 Object Lock for a bucket, review the requirements for S3 Object Lock buckets and objects and the lifecycle of objects in buckets with S3 Object Lock enabled.

Requirements for buckets with S3 Object Lock enabled

- If the global S3 Object Lock setting is enabled for the StorageGRID system, you can use the Tenant Manager, the Tenant Management API, or the S3 REST API to create buckets with S3 Object Lock enabled.

This example from the Tenant Manager shows a bucket with S3 Object Lock enabled.

Buckets

Create buckets and manage bucket settings.

1 bucket

Create bucket

Actions ▾

<input type="checkbox"/>	Name ▾	S3 Object Lock ? ▾	Region ▾	Object Count ? ▾	Space Used ? ▾	Date Created ▾
<input type="checkbox"/>	bank-records	✓	us-east-1	0	0 bytes	2021-01-06 16:53:19 MST

← Previous 1 Next →

- If you plan to use S3 Object Lock, you must enable S3 Object Lock when you create the bucket. You cannot enable S3 Object Lock for an existing bucket.
- Bucket versioning is required with S3 Object Lock. When S3 Object Lock is enabled for a bucket, StorageGRID automatically enables versioning for that bucket.
- After you create a bucket with S3 Object Lock enabled, you cannot disable S3 Object Lock or suspend versioning for that bucket.
- An StorageGRID bucket that has S3 Object Lock enabled does not have a default retention period. Instead, the S3 client application can optionally specify a retention date and legal hold setting for each object version that is added to that bucket.
- Bucket lifecycle configuration is supported for S3 Object Lifecycle buckets.
- CloudMirror replication is not supported for buckets with S3 Object Lock enabled.

Requirements for objects in buckets with S3 Object Lock enabled

- The S3 client application must specify retention settings for each object that needs to be protected by S3 Object Lock.
- You can increase the retain-until-date for an object version, but you can never decrease this value.
- If you are notified of a pending legal action or regulatory investigation, you can preserve relevant information by placing a legal hold on an object version. When an object version is under a legal hold, that object cannot be deleted from StorageGRID, even if it has reached its retain-until-date. As soon as the legal hold is lifted, the object version can be deleted if the retain-until-date has been reached.
- S3 Object Lock requires the use of versioned buckets. Retention settings apply to individual object versions. An object version can have both a retain-until-date and a legal hold setting, one but not the other, or neither. Specifying a retain-until-date or a legal hold setting for an object protects only the version specified in the request. You can create new versions of the object, while the previous version of the object remains locked.

Lifecycle of objects in buckets with S3 Object Lock enabled

Each object that is saved in a bucket with S3 Object Lock enabled goes through three stages:

1. Object ingest

- When adding an object version to a bucket with S3 Object Lock enabled, the S3 client application can optionally specify retention settings for the object (retain-until-date, legal hold, or both). StorageGRID

then generates metadata for that object, which includes a unique object identifier (UUID) and the ingest date and time.

- After an object version with retention settings is ingested, its data and S3 user-defined metadata cannot be modified.
- StorageGRID stores the object metadata independently of the object data. It maintains three copies of all object metadata at each site.

2. Object retention

- Multiple copies of the object are stored by StorageGRID. The exact number and type of copies and the storage locations are determined by the compliant rules in the active ILM policy.

3. Object deletion

- An object can be deleted when its retain-until-date is reached.
- An object that is under a legal hold cannot be deleted.

Creating an S3 bucket

You can use the Tenant Manager to create S3 buckets for object data. When you create a bucket, you must specify the bucket's name and region. If the global S3 Object Lock setting is enabled for the StorageGRID system, you can optionally enable S3 Object Lock for the bucket.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Manage All Buckets or the Root Access permission. These permissions override the permissions settings in group or bucket policies.
- If you plan to create a bucket with S3 Object Lock, the global S3 Object Lock setting must have been enabled for the StorageGRID system and you must have reviewed the requirements for S3 Object Lock buckets and objects.

[Using S3 Object Lock](#)

Steps

1. Select **STORAGE (S3) > Buckets**.

The Buckets page appears and lists any buckets that have already been created.

Buckets

Create buckets and manage bucket settings.

0 buckets Create bucket

Actions ▾

Name ▾	S3 Object Lock ? ▾	Region ▾	Object Count ? ▾	Space Used ? ▾	Date Created ▾
No buckets found					
Create bucket					

2. Select **Create bucket**.

The Create bucket wizard appears.

Create bucket

Enter bucket details

Enter the bucket's name and select the bucket's region.

Bucket name ?

Region ?

us-east-1 ▾

Cancel
Create bucket



If the global S3 Object Lock setting is enabled, Create bucket includes a second step for managing S3 Object Lock for the bucket.

3. Enter a unique name for the bucket.



You cannot change the bucket name after creating the bucket.

Bucket names must comply with these rules:

- Must be unique across each StorageGRID system (not just unique within the tenant account).
- Must be DNS compliant.
- Must contain at least 3 and no more than 63 characters.
- Can be a series of one or more labels, with adjacent labels separated by a period. Each label must start and end with a lowercase letter or a number and can only use lowercase letters, numbers, and hyphens.
- Must not look like a text-formatted IP address.
- Should not use periods in virtual hosted style requests. Periods will cause problems with server wildcard certificate verification.



See the Amazon Web Services (AWS) Documentation for more information.

4. Select the region for this bucket.

Your StorageGRID administrator manages the available regions. A bucket's region can affect the data-protection policy applied to objects. By default, all buckets are created in the `us-east-1` region.



You cannot change the region after creating the bucket.

5. Select **Create bucket** or **Continue**.

- If the global S3 Object Lock setting is not enabled, select **Create bucket**. The bucket is created and added to the table on the Buckets page.
- If the global S3 Object Lock setting is enabled, select **Continue**. Step 2, Manage S3 Object Lock, appears.

Create bucket

Enter details — 2 Manage S3 Object Lock (Optional)

Manage S3 Object Lock (This step is optional)

S3 Object Lock allows you to specify retention and legal hold settings for the objects ingested into a bucket. If you want to use S3 Object Lock, you must enable this setting when you create the bucket. You cannot add or disable S3 Object Lock after a bucket is created.

If S3 Object Lock is enabled, bucket versioning is required and will be enabled automatically.

☒ Enable S3 Object Lock

Previous **Create bucket**

6. Optionally, select the check box to enable S3 Object Lock for this bucket.

S3 Object Lock must be enabled for the bucket before an S3 client application can specify retain-until-date and legal hold settings for the objects added to the bucket.



You cannot enable or disable S3 Object Lock after creating the bucket.



If you enable S3 Object Lock for a bucket, bucket versioning is enabled automatically.

7. Select **Create bucket**.

The bucket is created and added to the table on the Buckets page.

Related information

[Manage objects with ILM](#)

[Understanding the Tenant Management API](#)

[Use S3](#)

Viewing S3 bucket details

You can view a list of the buckets and bucket settings in your tenant account.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.

Steps

1. Select **STORAGE (S3) > Buckets**.

The Buckets page appears and lists all buckets for the tenant account.

Buckets

Create buckets and manage bucket settings.

2 buckets Create bucket

Actions ▾

<input type="checkbox"/>	Name ▾	S3 Object Lock ? ▾	Region ▾	Object Count ? ▾	Space Used ? ▾	Date Created ▾
<input type="checkbox"/>	bucket-01	✓	us-east-1	0	0 bytes	2020-11-04 14:16:59 MST
<input type="checkbox"/>	bucket-02		us-east-1	0	0 bytes	2020-11-04 14:17:14 MST

← Previous **1** Next →

2. Review the information for each bucket.

As required, you can sort the information by any column, or you can page forward and back through the list.

- Name: The bucket's unique name, which cannot be changed.
- S3 Object Lock: Whether S3 Object Lock is enabled for this bucket.

This column is not displayed if the global S3 Object lock setting is disabled. This column also shows information for any legacy Compliant buckets.

- Region: The bucket's region, which cannot be changed.
- Object Count: The number of objects in this bucket.
- Space Used: The logical size of all objects in this bucket. The logical size does not include the actual space required for replicated or erasure-coded copies or for object metadata.
- Date Created: The date and time the bucket was created.



The Object Count and Space Used values displayed are estimates. These estimates are affected by the timing of ingests, network connectivity, and node status.

3. To view and manage the settings for a bucket, select the bucket name.

The bucket details page appears.

This page allows you to view and edit the settings for bucket options, bucket access, and platform services.

See the instructions for configuring each setting or platform service.

Buckets > bucket-02

Overview

Name: bucket-02

Region: us-east-1

S3 Object Lock: Disabled

Date created: 2020-11-04 14:51:59 MST

Bucket options

Bucket access

Platform services

Consistency level

Read-after-new-write

▼

Last access time updates

Disabled

▼

Related information

[Changing the consistency level](#)

[Enabling or disabling last access time updates](#)

[Configuring Cross-Origin Resource Sharing \(CORS\)](#)

[Configuring CloudMirror replication](#)

[Configuring event notifications](#)

[Configuring the search integration service](#)

Changing the consistency level

If you are using an S3 tenant, you can use the Tenant Manager or the Tenant Management API to change the consistency control for operations performed on the objects in S3 buckets.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Manage All Buckets or the Root Access permission. These permissions override the permissions settings in group or bucket policies.

About this task

Consistency level makes a trade-off between the availability of the objects and the consistency of those objects across different Storage Nodes and sites. In general, you should use the **Read-after-new-write** consistency level for your buckets. If the **Read-after-new-write** consistency level does not meet the client application's requirements, you can change the consistency level by setting the bucket consistency level or by using the `Consistency-Control` header. The `Consistency-Control` header overrides the bucket consistency level.



When you change a bucket's consistency level, only those objects that are ingested after the change are guaranteed to meet the revised level.

Steps

1. Select **STORAGE (S3) > Buckets**.
2. Select the bucket name from the list.

The bucket details page appears.

3. Select **Bucket options > Consistency level**.

Bucket options

Bucket access

Platform services

Consistency level

Read-after-new-write (default)

Change the consistency control for operations performed on the objects in the bucket. Consistency level makes a trade-off between the availability of the objects and the consistency of those objects across different Storage Nodes and sites.

In general, use the **Read-after-new-write** consistency level for your buckets. Then, if objects do not meet availability or consistency requirements, change the client application's behavior, or set the Consistency-Control header for an individual API request, which overrides the bucket setting.

☐

All

Provides the highest guarantee of consistency. All nodes receive the data immediately, or the request will fail.

☐

Strong-global

Guarantees read-after-write consistency for all client requests across all sites.

☐

Strong-site

Guarantees read-after-write consistency for all client requests within a site.

☒

Read-after-new-write (default)

Provides read-after-write consistency for new objects and eventual consistency for object updates. Offers high availability, and data protection guarantees.

Note: If your application attempts HEAD operations on keys that do not exist, set the Consistency Level to **Available**, unless you require AWS S3 consistency guarantees. Otherwise, a high number of 500 Internal Server errors can result if one or more Storage Nodes are unavailable.

☐

Available

Behaves the same as the **Read-after-new-write** consistency level, but only provides eventual consistency for HEAD operations. Offers higher availability for HEAD operations than **Read-after-new-write** if Storage Nodes are unavailable. Differs from AWS S3 consistency guarantees for HEAD operations only.

Save changes

4. Select a consistency level for operations performed on the objects in this bucket.

Consistency level	Description
All	All nodes receive the data immediately, or the request will fail.
Strong-global	Guarantees read-after-write consistency for all client requests across all sites.

Consistency level	Description
Strong-site	Guarantees read-after-write consistency for all client requests within a site.
Read-after-new-write (Default)	<p>Provides read-after-write consistency for new objects and eventual consistency for object updates. Offers high availability, and data protection guarantees. Matches Amazon S3 consistency guarantees.</p> <p>Note: If your application attempts HEAD operations on keys that do not exist, set the Consistency Level to Available, unless you require Amazon S3 consistency guarantees. Otherwise, a high number of 500 Internal Server errors can result if one or more Storage Nodes are unavailable.</p>
Available (eventual consistency for HEAD operations)	Behaves the same as the Read-after-new-write consistency level, but only provides eventual consistency for HEAD operations. Offers higher availability for HEAD operations than Read-after-new-write if Storage Nodes are unavailable. Differs from Amazon S3 consistency guarantees for HEAD operations only.

5. Select **Save changes**.

Related information

[Tenant management permissions](#)

Enabling or disabling last access time updates

When grid administrators create the information lifecycle management (ILM) rules for a StorageGRID system, they can optionally specify that an object's last access time be used to determine whether to move that object to a different storage location. If you are using an S3 tenant, you can take advantage of such rules by enabling last access time updates for the objects in an S3 bucket.

These instructions only apply to StorageGRID systems that include at least one ILM rule that uses the **Last Access Time** option in its placement instructions. You can ignore these instructions if your StorageGRID system does not include such a rule.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Manage All Buckets or the Root Access permission. These permissions override the permissions settings in group or bucket policies.

Last Access Time is one of the options available for the **Reference Time** placement instruction for an ILM rule. Setting the Reference Time for a rule to Last Access Time lets grid administrators specify that objects be placed in certain storage locations based on when those objects were last retrieved (read or viewed).

For example, to ensure that recently viewed objects remain on faster storage, a grid administrator can create an ILM rule specifying the following:

- Objects that have been retrieved in the past month should remain on local Storage Nodes.
- Objects that have not been retrieved in the past month should be moved to an off-site location.



See the instructions for managing objects with information lifecycle management.

By default, updates to last access time are disabled. If your StorageGRID system includes an ILM rule that uses the **Last Access Time** option and you want this option to apply to objects in this bucket, you must enable updates to last access time for the S3 buckets specified in that rule.



Updating the last access time when an object is retrieved can reduce StorageGRID performance, especially for small objects.

A performance impact occurs with last access time updates because StorageGRID must perform these additional steps every time objects are retrieved:

- Update the objects with new timestamps
- Add the objects to the ILM queue, so they can be reevaluated against current ILM rules and policy

The table summarizes the behavior applied to all objects in the bucket when last access time is disabled or enabled.

Type of request	Behavior if last access time is disabled (default)		Behavior if last access time is enabled	
	Last access time updated?	Object added to ILM evaluation queue?	Last access time updated?	Object added to ILM evaluation queue?
Request to retrieve an object, its access control list, or its metadata	No	No	Yes	Yes
Request to update an object's metadata	Yes	Yes	Yes	Yes
Request to copy an object from one bucket to another	<ul style="list-style-type: none"> • No, for the source copy • Yes, for the destination copy 	<ul style="list-style-type: none"> • No, for the source copy • Yes, for the destination copy 	<ul style="list-style-type: none"> • Yes, for the source copy • Yes, for the destination copy 	<ul style="list-style-type: none"> • Yes, for the source copy • Yes, for the destination copy
Request to complete a multipart upload	Yes, for the assembled object	Yes, for the assembled object	Yes, for the assembled object	Yes, for the assembled object

Steps

1. Select **STORAGE (S3) > Buckets**.
2. Select the bucket name from the list.

The bucket details page appears.

3. Select **Bucket options > Last access time updates**.
4. Select the appropriate radio button to enable or disable last access time updates.

Bucket options

Bucket access

Platform services

Consistency level

Read-after-new-write

▼

Last access time updates

Disabled

▲

Enable or disable last access time updates for the objects in this bucket.

When last access time updates are disabled, the following behavior applies to objects in the bucket:

- Requests to retrieve an object, its access control list, or its metadata do not update the object's last access time. The object is not added to ILM evaluation queues.
- Requests to update an object's metadata update the object's last access time. The object is added to ILM evaluation queues.
- Requests to copy an object from one bucket to another do not update the last access time for the source copy and do not add the source object to the ILM evaluation queue. However, the last access time is updated for the destination copy, and the destination object is added to ILM evaluation queues.
- A request to complete a multipart upload causes the last access time for the assembled object to be updated. The new object is added to ILM evaluation queues.

ⓘ

Updating the last access time when an object is retrieved can reduce performance, especially for small objects.

☐ Enable last access time updates when retrieving an object

☒ Disable last access time updates when retrieving an object

Save changes

5. Select **Save changes**.

Related information

[Tenant management permissions](#)

[Manage objects with ILM](#)

Configuring Cross-Origin Resource Sharing (CORS)

You can configure Cross-Origin Resource Sharing (CORS) for an S3 bucket if you want that bucket and objects in that bucket to be accessible to web applications in other domains.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Manage All Buckets or the Root Access permission. These permissions override the permissions settings in group or bucket policies.

About this task

Cross-Origin Resource Sharing (CORS) is a security mechanism that allows client web applications in one domain to access resources in a different domain. For example, suppose you use an S3 bucket named `Images` to store graphics. By configuring CORS for the `Images` bucket, you can allow the images in that bucket to be displayed on the website <http://www.example.com>.

Steps

1. Use a text editor to create the XML required to enable CORS.

This example shows the XML used to enable CORS for an S3 bucket. This XML allows any domain to send GET requests to the bucket, but it only allows the `http://www.example.com` domain to send POST and DELETE requests. All request headers are allowed.

```
<CORSConfiguration
  xmlns="http://s3.amazonaws.com/doc/2020-10-22/">
  <CORSRule>
    <AllowedOrigin>*</AllowedOrigin>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedHeader>*</AllowedHeader>
  </CORSRule>
  <CORSRule>
    <AllowedOrigin>http://www.example.com</AllowedOrigin>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>
    <AllowedHeader>*</AllowedHeader>
  </CORSRule>
</CORSConfiguration>
```

For more information about the CORS configuration XML, see [Amazon Web Services \(AWS\) Documentation: Amazon Simple Storage Service Developer Guide](#).

2. In the Tenant Manager, select **STORAGE (S3) > Buckets**.
3. Select the bucket name from the list.

The bucket details page appears.

4. Select **Bucket access > Cross-Origin Resource Sharing (CORS)**.
5. Select the **Enable CORS** check box.
6. Paste the CORS configuration XML into the text box, and select **Save changes**.

Bucket options

Bucket access

Platform services

Cross-Origin Resource Sharing (CORS)

Disabled

Configure Cross-Origin Resource Sharing (CORS) for an S3 bucket if you want that bucket and objects in that bucket to be accessible to web applications in other domains.

☒ Enable CORS

Clear

```

<CORSConfiguration
  xmlns="http://s3.amazonaws.com/doc/2020-10-22/">
  <CORSRule>
    <AllowedOrigin>*</AllowedOrigin>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedHeader>*</AllowedHeader>
  </CORSRule>
  <CORSRule>
    <AllowedOrigin>http://www.example.com</AllowedOrigin>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>
  </CORSRule>
</CORSConfiguration>

```

Save changes

- To modify the CORS setting for the bucket, update the CORS configuration XML in the text box or select **Clear** to start over. Then select **Save changes**.
- To disable CORS for the bucket, unselect the **Enable CORS** check box, and then select **Save changes**.

Deleting an S3 bucket

You can use the Tenant Manager to delete an S3 bucket that is empty.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Manage All Buckets or the Root Access permission. These permissions override the permissions settings in group or bucket policies.

About this task

These instructions describe how to delete an S3 bucket using the Tenant Manager. You can also delete S3 buckets using the Tenant Management API or the S3 REST API.

You cannot delete an S3 bucket if it contains objects or noncurrent object versions. For information about how S3 versioned objects are deleted, see the instructions for managing objects with information lifecycle management.

Steps

1. Select **STORAGE (S3) > Buckets**.

The Buckets page appears and shows all existing S3 buckets.

Buckets

Create buckets and manage bucket settings.

2 buckets Create bucket

Actions

<input type="checkbox"/>	Name	S3 Object Lock	Region	Object Count	Space Used	Date Created
<input type="checkbox"/>	bucket-01	✓	us-east-1	0	0 bytes	2020-11-04 14:16:59 MST
<input type="checkbox"/>	bucket-02		us-east-1	0	0 bytes	2020-11-04 14:17:14 MST

< Previous 1 Next >

2. Select the check box for the empty bucket you want to delete.

The Actions menu is enabled.

3. From the Actions menu, select **Delete empty bucket**.

Actions

Delete empty bucket

<input type="checkbox"/>	Name	S3 Object Lock	Region	Object Count	Space Used	Date Created
<input type="checkbox"/>	bucket-01	✓	us-east-1	0	0 bytes	2020-11-04 14:16:59 MST
<input checked="" type="checkbox"/>	bucket-02		us-east-1	0	0 bytes	2020-11-04 14:17:14 MST

A confirmation message appears.

Delete empty bucket

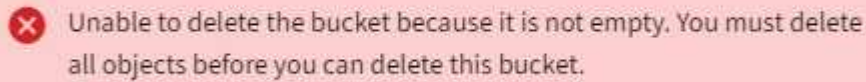
Are you sure you want to delete empty bucket bucket-02?

Cancel Delete bucket

4. If you are sure you want to delete the bucket, select **Delete bucket**.

StorageGRID confirms that the bucket is empty and then deletes the bucket. This operation might take a few minutes.

If the bucket is not empty, an error message appears. You must delete all objects before you can delete the bucket.

A red rectangular box with a white border. On the left is a red circle with a white 'X'. To its right, the text reads: "Unable to delete the bucket because it is not empty. You must delete all objects before you can delete this bucket."

Related information

[Manage objects with ILM](#)

Managing S3 platform services

If the use of platform services is allowed for your S3 tenant account, you can use platform services to leverage external services and configure CloudMirror replication, notifications, and search integration for S3 buckets.

- [What platform services are](#)
- [Considerations for using platform services](#)
- [Configuring platform services endpoints](#)
- [Configuring CloudMirror replication](#)
- [Configuring event notifications](#)
- [Using the search integration service](#)

What platform services are

StorageGRID platform services can help you implement a hybrid cloud strategy.

If the use of platform services is allowed for your tenant account, you can configure the following services for any S3 bucket:

- **CloudMirror replication:** The StorageGRID CloudMirror replication service is used to mirror specific objects from a StorageGRID bucket to a specified external destination.

For example, you might use CloudMirror replication to mirror specific customer records into Amazon S3 and then leverage AWS services to perform analytics on your data.



CloudMirror replication is not supported if the source bucket has S3 Object Lock enabled.

- **Notifications:** Per-bucket event notifications are used to send notifications about specific actions performed on objects to a specified external Amazon Simple Notification Service™ (SNS).

For example, you could configure alerts to be sent to administrators about each object added to a bucket, where the objects represent log files associated with a critical system event.



Although event notification can be configured on a bucket with S3 Object Lock enabled, the S3 Object Lock metadata (including Retain Until Date and Legal Hold status) of the objects will not be included in the notification messages.

- **Search integration service:** The search integration service is used to send S3 object metadata to a specified Elasticsearch index where the metadata can be searched or analyzed using the external service.

For example, you could configure your buckets to send S3 object metadata to a remote Elasticsearch service. You could then use Elasticsearch to perform searches across buckets, and perform sophisticated analyses of patterns present in your object metadata.



Although Elasticsearch integration can be configured on a bucket with S3 Object Lock enabled, the S3 Object Lock metadata (including Retain Until Date and Legal Hold status) of the objects will not be included in the notification messages.

Because the target location for platform services is typically external to your StorageGRID deployment, platform services give you the power and flexibility that comes from using external storage resources, notification services, and search or analysis services for your data.

Any combination of platform services can be configured for a single S3 bucket. For example, you could configure both the CloudMirror service and notifications on a StorageGRID S3 bucket so that you can mirror specific objects to the Amazon Simple Storage Service, while sending a notification about each such object to a third party monitoring application to help you track your AWS expenses.



The use of platform services must be enabled for each tenant account by a StorageGRID administrator using the Grid Manager or the Grid Management API.

How platform services are configured

Platform services communicate with external endpoints that you configure using the Tenant Manager or the Tenant Management API. Each endpoint represents an external destination, such as a StorageGRID S3 bucket, an Amazon Web Services bucket, a Simple Notification Service (SNS) topic, or an Elasticsearch cluster hosted locally, on AWS, or elsewhere.

After you create an endpoint, you can enable a platform service for a bucket by adding XML configuration to the bucket. The XML configuration identifies the objects that the bucket should act on, the action that the bucket should take, and the endpoint that the bucket should use for the service.

You must add separate XML configurations for each platform service that you want to configure. For example:

1. If you want all objects whose keys start with `/images` to be replicated to an Amazon S3 bucket, you must add a replication configuration to the source bucket.
2. If you also want to send notifications when these objects are stored to the bucket, you must add a notifications configuration.
3. Finally, if you want to index the metadata for these objects, you must add the metadata notification configuration that is used to implement search integration.

The format for the configuration XML is governed by the S3 REST APIs used to implement StorageGRID platform services:

Platform service	S3 REST API
CloudMirror replication	<ul style="list-style-type: none"> • GET Bucket replication • PUT Bucket replication
Notifications	<ul style="list-style-type: none"> • GET Bucket notification • PUT Bucket notification
Search integration	<ul style="list-style-type: none"> • GET Bucket metadata notification configuration • PUT Bucket metadata notification configuration <p>These operations are custom to StorageGRID.</p>

See the instructions for implementing S3 client applications for details on how StorageGRID implements these APIs.

Related information

[Use S3](#)

[Understanding the CloudMirror replication service](#)

[Understanding notifications for buckets](#)

[Understanding the search integration service](#)

[Considerations for using platform services](#)

Understanding the CloudMirror replication service

You can enable CloudMirror replication for an S3 bucket if you want StorageGRID to replicate specified objects added to the bucket to one or more destination buckets.

CloudMirror replication operates independently of the grid's active ILM policy. The CloudMirror service replicates objects as they are stored to the source bucket and delivers them to the destination bucket as soon as possible. Delivery of replicated objects is triggered when object ingest succeeds.

If you enable CloudMirror replication for an existing bucket, only the new objects added to that bucket are replicated. Any existing objects in the bucket are not replicated. To force the replication of existing objects, you can update the existing object's metadata by performing an object copy.



If you are using CloudMirror replication to copy objects to an AWS S3 destination, be aware that Amazon S3 limits the size of user-defined metadata within each PUT request header to 2 KB. If an object has user-defined metadata greater than 2 KB, that object will not be replicated.

In StorageGRID, you can replicate the objects in a single bucket to multiple destination buckets. To do so, specify the destination for each rule in the replication configuration XML. You cannot replicate an object to more than one bucket at the same time.

Additionally, you can configure CloudMirror replication on versioned or unversioned buckets, and you can specify a versioned or unversioned bucket as the destination. You can use any combination of versioned and unversioned buckets. For example, you could specify a versioned bucket as the destination for an unversioned

source bucket, or vice versa. You can also replicate between unversioned buckets.

Deletion behavior for the CloudMirror replication service is the same as the deletion behavior of the Cross Region Replication (CRR) service provided by Amazon S3 — deleting an object in a source bucket never deletes a replicated object in the destination. If both source and destination buckets are versioned, the delete marker is replicated. If the destination bucket is not versioned, deleting an object in the source bucket does not replicate the delete marker to the destination bucket or delete the destination object.

As objects are replicated to the destination bucket, StorageGRID marks them as “replicas.” A destination StorageGRID bucket will not replicate objects marked as replicas again, protecting you from accidental replication loops. This replica marking is internal to StorageGRID and does not prevent you from leveraging AWS CRR when using an Amazon S3 bucket as the destination.



The custom header used to mark a replica is `x-ntap-sg-replica`. This marking prevents a cascading mirror. StorageGRID does support a bi-directional CloudMirror between two grids.

The uniqueness and ordering of events in the destination bucket are not guaranteed. More than one identical copy of a source object might be delivered to the destination as a result of operations taken to guarantee delivery success. In rare cases, when the same object is updated simultaneously from two or more different StorageGRID sites, the ordering of operations on the destination bucket might not match the ordering of events on the source bucket.

CloudMirror replication is typically configured to use an external S3 bucket as a destination. However, you can also configure replication to use another StorageGRID deployment or any S3-compatible service.

Related information

[Configuring CloudMirror replication](#)

Understanding notifications for buckets

You can enable event notification for an S3 bucket if you want StorageGRID to send notifications about specified events to a destination Amazon Simple Notification Service (SNS).

You can configure event notifications by associating notification configuration XML with a source bucket. The notification configuration XML follows S3 conventions for configuring bucket notifications, with the destination SNS topic specified as the URN of an endpoint.

Event notifications are created at the source bucket as specified in the notification configuration and are delivered to the destination. If an event associated with an object succeeds, a notification about that event is created and queued for delivery.

The uniqueness and ordering of notifications are not guaranteed. More than one notification of an event might be delivered to the destination as a result of operations taken to guarantee delivery success. And because delivery is asynchronous, the time ordering of notifications at the destination is not guaranteed to match the ordering of events on the source bucket, particularly for operations that originate from different StorageGRID sites. You can use the `sequencer` key in the event message to determine the order of events for a particular object, as described in Amazon S3 documentation.

Supported notifications and messages

StorageGRID event notification follows the Amazon S3 API with the following limitations:

- You cannot configure a notification for the following event types. These event types are **not** supported.
 - `s3:ReducedRedundancyLostObject`
 - `s3:ObjectRestore:Completed`
- Event notifications sent from StorageGRID use the standard JSON format except that they do not include some keys and use specific values for others, as shown in the table:

Key name	StorageGRID value
eventSource	sgws:s3
awsRegion	not included
x-amz-id-2	not included
arn	urn:sgws:s3:::bucket_name

Related information

[Configuring event notifications](#)

Understanding the search integration service

You can enable search integration for an S3 bucket if you want to use an external search and data analysis service for your object metadata.

The search integration service is a custom StorageGRID service that automatically and asynchronously sends S3 object metadata to a destination endpoint whenever an object or its metadata is updated. You can then use sophisticated search, data analysis, visualization, or machine learning tools provided by the destination service to search, analyze, and gain insights from your object data.

You can enable the search integration service for any versioned or unversioned bucket. Search integration is configured by associating metadata notification configuration XML with the bucket that specifies which objects to act on and the destination for the object metadata.

Notifications are generated in the form of a JSON document named with the bucket name, object name, and version ID, if any. Each metadata notification contains a standard set of system metadata for the object in addition to all of the object's tags and user metadata.



For tags and user metadata, StorageGRID passes dates and numbers to Elasticsearch as strings or as S3 event notifications. To configure Elasticsearch to interpret these strings as dates or numbers, follow the Elasticsearch instructions for dynamic field mapping and for mapping date formats. You must enable the dynamic field mappings on the index before you configure the search integration service. After a document is indexed, you cannot edit the document's field types in the index.

Notifications are generated and queued for delivery whenever:

- An object is created.
- An object is deleted, including when objects are deleted as a result of the operation of the grid's ILM policy.

- Object metadata or tags are added, updated, or deleted. The complete set of metadata and tags is always sent on update — not just the changed values.

After you add metadata notification configuration XML to a bucket, notifications are sent for any new objects that you create and for any objects that you modify by updating its data, user metadata, or tags. However, notifications are not sent for any objects that were already in the bucket. To ensure that object metadata for all objects in the bucket is sent to the destination, you should do either of the following:

- Configure the search integration service immediately after creating the bucket and before adding any objects.
- Perform an action on all objects already in the bucket that will trigger a metadata notification message to be sent to the destination.

The StorageGRID search integration service supports an Elasticsearch cluster as a destination. As with the other platform services, the destination is specified in the endpoint whose URN is used in the configuration XML for the service. Use the *Interoperability Matrix Tool* to determine the supported versions of Elasticsearch.

Related information

[NetApp Interoperability Matrix Tool](#)

[Configuration XML for search integration](#)

[Object metadata included in metadata notifications](#)

[JSON generated by the search integration service](#)

[Configuring the search integration service](#)

Considerations for using platform services

Before implementing platform services, review the recommendations and considerations for using these services.

Considerations for using platform services

Consideration	Details
Destination endpoint monitoring	You must monitor the availability of each destination endpoint. If connectivity to the destination endpoint is lost for an extended period of time and a large backlog of requests exists, additional client requests (such as PUT requests) to StorageGRID will fail. You must retry these failed requests when the endpoint becomes reachable.

Consideration	Details
Destination endpoint throttling	<p>StorageGRID software might throttle incoming S3 requests for a bucket if the rate at which the requests are being sent exceeds the rate at which the destination endpoint can receive the requests. Throttling only occurs when there is a backlog of requests waiting to be sent to the destination endpoint.</p> <p>The only visible effect is that the incoming S3 requests will take longer to execute. If you start to detect significantly slower performance, you should reduce the ingest rate or use an endpoint with higher capacity. If the backlog of requests continues to grow, client S3 operations (such as PUT requests) will eventually fail.</p> <p>CloudMirror requests are more likely to be affected by the performance of the destination endpoint because these requests typically involve more data transfer than search integration or event notification requests.</p>
Ordering guarantees	<p>StorageGRID guarantees ordering of operations on an object within a site. As long as all operations against an object are within the same site, the final object state (for replication) will always equal the state in StorageGRID.</p> <p>StorageGRID makes a best effort attempt to order requests when operations are made across StorageGRID sites. For example, if you write an object initially to site A and then later overwrite the same object at site B, the final object replicated by CloudMirror to the destination bucket is not guaranteed to be the newer object.</p>
ILM-driven object deletions	<p>To match the deletion behavior of the AWS CRR and SNS services, CloudMirror and event notification requests are not sent when an object in the source bucket is deleted because of StorageGRID ILM rules. For example, no CloudMirror or event notifications requests are sent if an ILM rule deletes an object after 14 days.</p> <p>In contrast, search integration requests are sent when objects are deleted because of ILM.</p>

Considerations for using the CloudMirror replication service

Consideration	Details
Replication status	StorageGRID does not support the <code>x-amz-replication-status</code> header.
Object size	The maximum size for objects that can be replicated to a destination bucket by the CloudMirror replication service is 5 TB, which is the same as the maximum object size supported by StorageGRID.

Bucket versioning and version IDs	<p>If the source S3 bucket in StorageGRID has versioning enabled, you should also enable versioning for the destination bucket.</p> <p>When using versioning, note that the ordering of object versions in the destination bucket is best effort and not guaranteed by the CloudMirror service, due to limitations in the S3 protocol.</p> <p>Note: Version IDs for the source bucket in StorageGRID are not related to the version IDs for the destination bucket.</p>
Tagging for object versions	<p>The CloudMirror service does not replicate any PUT Object tagging or DELETE Object tagging requests that supply a version ID, due to limitations in the S3 protocol. Because version IDs for the source and destination are not related, there is no way to ensure that a tag update to a specific version ID will be replicated.</p> <p>In contrast, the CloudMirror service does replicate PUT Object tagging requests or DELETE Object tagging requests that do not specify a version ID. These requests update the tags for the latest key (or the latest version if the bucket is versioned). Normal ingests with tags (not tagging updates) are also replicated.</p>
Multipart uploads and ETag values	<p>When mirroring objects that were uploaded using a multipart upload, the CloudMirror service does not preserve the parts. As a result, the ETag value for the mirrored object will be different than the ETag value of the original object.</p>
Objects encrypted with SSE-C (server-side encryption with customer-provided keys)	<p>The CloudMirror service does not support objects that are encrypted with SSE-C. If you attempt to ingest an object into the source bucket for CloudMirror replication and the request includes the SSE-C request headers, the operation fails.</p>
Bucket with S3 Object Lock enabled	<p>If the destination S3 bucket for CloudMirror replication has S3 Object Lock enabled, the replication operation will fail with an AccessDenied error.</p>

Related information

[Use S3](#)

Configuring platform services endpoints

Before you can configure a platform service for a bucket, you must configure at least one

endpoint to be the destination for the platform service.

Access to platform services is enabled on a per-tenant basis by a StorageGRID administrator. To create or use a platform services endpoint, you must be a tenant user with Manage Endpoints or Root Access permission, in a grid whose networking has been configured to allow Storage Nodes to access external endpoint resources. Contact your StorageGRID administrator for more information.

What a platform services endpoint is

When you create a platform services endpoint, you specify the information that StorageGRID needs to access the external destination.

For example, if you want to replicate objects from a StorageGRID bucket to an S3 bucket, you create a platform services endpoint that includes the information and credentials StorageGRID needs to access the destination bucket on AWS.

Each type of platform service requires its own endpoint, so you must configure at least one endpoint for each platform service you plan to use. After defining a platform services endpoint, you use the endpoint's URN as the destination in the configuration XML used to enable the service.

You can use the same endpoint as the destination for more than one source bucket. For example, you could configure several source buckets to send object metadata to the same search integration endpoint so that you can perform searches across multiple buckets. You can also configure a source bucket to use more than one endpoint as a target, which enables you to do things like send notifications about object creation to one SNS topic and notifications about object deletion to a second SNS topic.

Endpoints for CloudMirror replication

StorageGRID supports replication endpoints that represent S3 buckets. These buckets might be hosted on Amazon Web Services, the same or a remote StorageGRID deployment, or another service.

Endpoints for notifications

StorageGRID supports Simple Notification Service (SNS) endpoints. Simple Queue Service (SQS) or AWS Lambda endpoints are not supported.

Endpoints for the search integration service

StorageGRID supports search integration endpoints that represent Elasticsearch clusters. These Elasticsearch clusters can be in a local datacenter or hosted in an AWS cloud or elsewhere.

The search integration endpoint refers to a specific Elasticsearch index and type. You must create the index in Elasticsearch before creating the endpoint in StorageGRID, or endpoint creation will fail. You do not need to create the type before creating the endpoint. StorageGRID will create the type if required when it sends object metadata to the endpoint.

Related information

[Administer StorageGRID](#)

Specifying the URN for a platform services endpoint

When you create a platform services endpoint, you must specify a Unique Resource Name (URN). You will use the URN to reference the endpoint when you create configuration XML for the platform service. The URN for each endpoint must be unique.

StorageGRID validates platform services endpoints as you create them. Before you create a platform services endpoint, confirm that the resource specified in the endpoint exists and that it can be reached.

URN elements

The URN for a platform services endpoint must start with either `arn:aws` or `urn:mystore`, as follows:

- If the service is hosted on AWS, use `arn:aws`.
- If the service is hosted locally, use `urn:mystore`

For example, if you are specifying the URN for a CloudMirror endpoint hosted on StorageGRID, the URN might begin with `urn:sgws`.

The next element of the URN specifies the type of platform service, as follows:

Service	Type
CloudMirror replication	s3
Notifications	sns
Search integration	es

For example, to continue specifying the URN for a CloudMirror endpoint hosted on StorageGRID, you would add `s3` to get `urn:sgws:s3`.

The final element of the URN identifies the specific target resource at the destination URI.

Service	Specific resource
CloudMirror replication	bucket-name
Notifications	sns-topic-name
Search integration	domain-name/index-name/type-name Note: If the Elasticsearch cluster is not configured to create indexes automatically, you must create the index manually before you create the endpoint.

URNs for services hosted on AWS

For AWS entities, the complete URN is a valid AWS ARN. For example:

- CloudMirror replication:

```
arn:aws:s3:::bucket-name
```

- Notifications:

```
arn:aws:sns:region:account-id:topic-name
```

- Search integration:

```
arn:aws:es:region:account-id:domain/domain-name/index-name/type-name
```



For an AWS search integration endpoint, the `domain-name` must include the literal string `domain/`, as shown here.

URNs for locally-hosted services

When using locally-hosted services instead of cloud services, you can specify the URN in any way that creates a valid and unique URN, as long as the URN includes the required elements in the third and final positions. You can leave the elements indicated by optional blank, or you can specify them in any way that helps you identify the resource and make the URN unique. For example:

- CloudMirror replication:

```
urn:mysite:s3:optional:optional:bucket-name
```

For a CloudMirror endpoint hosted on StorageGRID, you can specify a valid URN that begins with `urn:sgws:`

```
urn:sgws:s3:optional:optional:bucket-name
```

- Notifications:

```
urn:mysite:sns:optional:optional:sns-topic-name
```

- Search integration:

```
urn:mysite:es:optional:optional:domain-name/index-name/type-name
```



For locally-hosted search integration endpoints, the `domain-name` element can be any string as long as the URN of the endpoint is unique.

Creating a platform services endpoint

You must create at least one endpoint of the correct type before you can enable a

platform service.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- Platform services must be enabled for your tenant account by a StorageGRID administrator.
- You must belong to a user group that has the Manage Endpoints permission.
- The resource referenced by the platform services endpoint must have been created:
 - CloudMirror replication: S3 bucket
 - Event notification: SNS topic
 - Search notification: Elasticsearch index, if the destination cluster is not configured to automatically create indexes.
- You must have the information about the destination resource:
 - Host and port for the Uniform Resource Identifier (URI)



If you plan to use a bucket hosted on a StorageGRID system as an endpoint for CloudMirror replication, contact the grid administrator to determine the values you need to enter.

- Unique Resource Name (URN)

Specifying the URN for a platform services endpoint

- Authentication credentials (if required):
 - Access Key: Access key ID and secret access key
 - Basic HTTP: Username and password
- Security certificate (if using a custom CA certificate)

Steps

1. Select **STORAGE (S3) > Platform services endpoints**.

The Platform services endpoints page appears.

Platform services endpoints

A platform services endpoint stores the information StorageGRID needs to use an external resource as a target for a platform service (CloudMirror replication, notifications, or search integration). You must configure an endpoint for each platform service you plan to use.

0 endpoints

Create endpoint

Delete endpoint

	Display name ?	Last error ?	Type ?	URI ?	URN ?
No endpoints found					
Create endpoint					

2. Select **Create endpoint**.

Create endpoint

1 Enter details

2 Select authentication type
Optional

3 Verify server
Optional

Enter endpoint details

Enter the endpoint's display name, URI, and URN.

Display name ?

URI ?

https://example.com

URN ?

arn:aws:s3::bucket_name

Cancel

Continue

3. Enter a display name to briefly describe the endpoint and its purpose.

The type of platform service that the endpoint supports is shown beside the endpoint name when it is listed on the Endpoints page, so you do not need to include that information in the name.

4. In the **URI** field, specify the Unique Resource Identifier (URI) of the endpoint.

Use one of the following formats:

```
https://host:port
http://host:port
```

If you do not specify a port, port 443 is used for HTTPS URIs and port 80 is used for HTTP URIs.

For example, the URI for a bucket hosted on StorageGRID might be:

```
https://s3.example.com:10443
```

In this example, `s3.example.com` represents the DNS entry for the virtual IP (VIP) of the StorageGRID high availability (HA) group, and `10443` represents the port defined in the load balancer endpoint.



Whenever possible, you should connect to a HA group of load-balancing nodes to avoid a single point of failure.

Similarly, the URI for a bucket hosted on AWS might be:

```
https://s3-aws-region.amazonaws.com
```



If the endpoint is used for the CloudMirror replication service, do not include the bucket name in the URI. You include the bucket name in the **URN** field.

5. Enter the Unique Resource Name (URN) for the endpoint.



You cannot change an endpoint's URN after the endpoint has been created.

6. Select **Continue**.

7. Select a value for **Authentication type**, and then enter the required credentials.

Create endpoint

1 Enter details — 2 Select authentication type Optional — 3 Verify server Optional

Authentication type ?

Select the method used to authenticate connections to the endpoint.

Anonymous ✓

Anonymous

Access Key

Basic HTTP

Previous Continue

The credentials that you supply must have write permissions for the destination resource.

Authentication type	Description	Credentials
Anonymous	Provides anonymous access to the destination. Only works for endpoints that have security disabled.	No authentication.
Access Key	Uses AWS-style credentials to authenticate connections with the destination.	<ul style="list-style-type: none"> • Access key ID • Secret access key
Basic HTTP	Uses a username and password to authenticate connections to the destination.	<ul style="list-style-type: none"> • Username • Password

8. Select **Continue**.

9. Select a radio button for **Verify server** to choose how TLS connection to the endpoint is verified.

×

Create endpoint

✓ Enter details

✓ Select authentication type
Optional

3 Verify server
Optional

Verify server

Use this method to validate the certificate for TLS connections to the endpoint resource. If you select "Use custom CA certificate," copy and paste the custom security certificate in the text box.

☒ Use custom CA certificate
 ☐ Use operating system CA certificate
 ☐ Do not verify certificate

```

-----BEGIN CERTIFICATE-----
abcdefghijklmnopqrstuvwxyz123456780ABCDEFGHIJKL
123456/7890ABCDEFabcdefghijklmnopqrstuvwxyzABCD
-----END CERTIFICATE-----

```

Previous

Test and create endpoint

Type of certificate verification	Description
Use custom CA certificate	Use a custom security certificate. If you select this setting, copy and paste the custom security certificate in the CA Certificate text box.

Type of certificate verification	Description
Use operating system CA certificate	Use the default CA certificate installed on the operating system to secure connections.
Do not verify certificate	The certificate used for the TLS connection is not verified. This option is not secure.

10. Select **Test and create endpoint**.

- A success message appears if the endpoint can be reached using the specified credentials. The connection to the endpoint is validated from one node at each site.
- An error message appears if endpoint validation fails. If you need to modify the endpoint to correct the error, select **Return to endpoint details** and update the information. Then, select **Test and create endpoint**.



Endpoint creation fails if platform services are not enabled for your tenant account. Contact your StorageGRID administrator.

After you have configured an endpoint, you can use its URN to configure a platform service.

Related information

[Specifying the URN for a platform services endpoint](#)

[Configuring CloudMirror replication](#)

[Configuring event notifications](#)

[Configuring the search integration service](#)

Testing the connection for a platform services endpoint

If the connection to a platform service has changed, you can test the connection for the endpoint to validate that the destination resource exists and that it can be reached using the credentials you specified.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Manage Endpoints permission.

About this task

StorageGRID does not validate that the credentials have the correct permissions.

Steps

1. Select **STORAGE (S3) > Platform services endpoints**.

The Platform services endpoints page appears and shows the list of platform services endpoints that have already been configured.







Platform services endpoints

A platform services endpoint stores the information StorageGRID needs to use an external resource as a target for a platform service (CloudMirror replication, notifications, or search integration). You must configure an endpoint for each platform service you plan to use.

4 endpoints

Create endpoint


Delete endpoint

<input type="checkbox"/>	Display name 	Last error 	Type 	URI 	URN 
<input type="checkbox"/>	my-endpoint-1		S3 Bucket	http://10.96.104.167:10443	urn:sgws:s3:::bucket1
<input type="checkbox"/>	my-endpoint-2	 2 hours ago	Search	http://10.96.104.30:9200	urn:sgws:es:::mydomain/sveloso/_doc
<input type="checkbox"/>	my-endpoint-3		Notifications	http://10.96.104.202:8080/	arn:aws:sns:us-west-2::example1
<input type="checkbox"/>	my-endpoint-4		S3 Bucket	http://10.96.104.167:10443	urn:sgws:s3:::bucket2

2. Select the endpoint whose connection you want to test.

The endpoint details page appears.

Overview

Display name: **my-endpoint-1** 

Type: **S3 Bucket**

URI: **http://10.96.104.167:10443**

URN: **urn:sgws:s3:::bucket1**

Connection

Configuration

Verify connection

Some errors might continue to appear after they are resolved. To see if an error is current or to force the removal of a resolved error, select **Test connection**.

Test connection

3. Select **Test connection**.

- A success message appears if the endpoint can be reached using the specified credentials. The connection to the endpoint is validated from one node at each site.
- An error message appears if endpoint validation fails. If you need to modify the endpoint to correct the error, select **Configuration** and update the information. Then, select **Test and save changes**.

Editing a platform services endpoint

You can edit the configuration for a platform services endpoint to change its name, URI, or other details. For example, you might need to update expired credentials or change the URI to point to a backup Elasticsearch index for failover. You cannot change the URN for a platform services endpoint.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the Manage Endpoints permission.

Steps

1. Select **STORAGE (S3) > Platform services endpoints**.

The Platform services endpoints page appears and shows the list of platform services endpoints that have already been configured.

Platform services endpoints

A platform services endpoint stores the information StorageGRID needs to use an external resource as a target for a platform service (CloudMirror replication, notifications, or search integration). You must configure an endpoint for each platform service you plan to use.

4 endpoints Create endpoint

Delete endpoint

<input type="checkbox"/>	Display name	Last error	Type	URI	URN
<input type="checkbox"/>	my-endpoint-1		S3 Bucket	http://10.96.104.167:10443	urn:sgws:s3:::bucket1
<input type="checkbox"/>	my-endpoint-2	2 hours ago	Search	http://10.96.104.30:9200	urn:sgws:es:::mydomain/sveloso/_doc
<input type="checkbox"/>	my-endpoint-3		Notifications	http://10.96.104.202:8080/	arn:aws:sns:us-west-2::example1
<input type="checkbox"/>	my-endpoint-4		S3 Bucket	http://10.96.104.167:10443	urn:sgws:s3:::bucket2

2. Select the endpoint you want to edit.

The endpoint details page appears.

3. Select **Configuration**.

Overview

Display name: **my-endpoint-3** 

Type: **Notifications**

URI: **http://10.96.104.202:8080/**

URN: **arn:aws:sns:us-west-2::example1**

Connection

Configuration

Edit configuration

Endpoint details

URI 

http://10.96.104.202:8080/

URN 

arn:aws:sns:us-west-2::example1

Authentication type

Basic HTTP 

Username 

testme

Password 

••••••••

Edit password

Verify server

- ☐ Use custom CA certificate
- ☒ Use operating system CA certificate
- ☐ Do not verify certificate

```
-----BEGIN CERTIFICATE-----
abcdefghijklmnopqrstuvwxyz
123456/7890ABCDEFabcdefghijklmnopqrstuvwxyz
-----END CERTIFICATE-----
```

Test and save changes

4. As needed, change the configuration of the endpoint.



You cannot change an endpoint's URN after the endpoint has been created.

a. To change the display name for the endpoint, select the edit icon .

b. As needed, change the URI.

c. As needed, change the authentication type.

- For Basic HTTP authentication, change the username as needed. Change the password as needed by selecting **Edit password** and entering the new password. If you need to cancel your changes, select **Revert password edit**.
- For Access Key authentication, change the key as necessary by selecting **Edit S3 key** and pasting a new access key ID and secret access key. If you need to cancel your changes, select **Revert S3 key edit**.

d. As needed, change the method for verifying the server.

5. Select **Test and save changes**.

- A success message appears if the endpoint can be reached using the specified credentials. The connection to the endpoint is verified from one node at each site.
- An error message appears if endpoint validation fails. Modify the endpoint to correct the error, and then select **Test and save changes**.

Related information

[Creating a platform services endpoint](#)

Deleting a platform services endpoint

You can delete an endpoint if you no longer want to use the associated platform service.

What you'll need

- You must be signed in to the Tenant Manager using a supported browser.
- You must belong to a user group that has the **Manage Endpoints** permission.

Steps

1. Select **STORAGE (S3) > Platform services endpoints**.

The Platform services endpoints page appears and shows the list of platform services endpoints that have already been configured.







Platform services endpoints

A platform services endpoint stores the information StorageGRID needs to use an external resource as a target for a platform service (CloudMirror replication, notifications, or search integration). You must configure an endpoint for each platform service you plan to use.

4 endpoints

Create endpoint

Delete endpoint

<input type="checkbox"/>	Display name 	Last error 	Type 	URI 	URN 
<input type="checkbox"/>	my-endpoint-1		S3 Bucket	http://10.96.104.167:10443	urn:sgws:s3:::bucket1
<input type="checkbox"/>	my-endpoint-2	 2 hours ago	Search	http://10.96.104.30:9200	urn:sgws:es:::mydomain/sveloso/_doc
<input type="checkbox"/>	my-endpoint-3		Notifications	http://10.96.104.202:8080/	arn:aws:sns:us-west-2::example1
<input type="checkbox"/>	my-endpoint-4		S3 Bucket	http://10.96.104.167:10443	urn:sgws:s3:::bucket2

2. Select the check box for each endpoint you want to delete.



If you delete a platform services endpoint that is in use, the associated platform service will be disabled for any buckets that use the endpoint. Any requests that have not yet been completed will be dropped. Any new requests will continue to be generated until you change your bucket configuration to no longer reference the deleted URN. StorageGRID will report these requests as unrecoverable errors.

3. Select **Actions** > **Delete endpoint**.

A confirmation message appears.

Delete endpoint

Are you sure you want to delete endpoint my-endpoint-10?

This might take a few minutes.

When you delete an endpoint, you can no longer use it to access external resources.

Cancel

Delete endpoint

4. Select **Delete endpoint**.

Troubleshooting platform services endpoint errors

If an error occurs when StorageGRID attempts to communicate with a platform services endpoint, a message is displayed on the Dashboard. On the Platform services endpoints page, the Last error column indicates how long ago the error occurred. No error is displayed if the permissions associated with an endpoint's credentials are incorrect.

Determining if an error has occurred


If any platform services endpoint errors have occurred within the past 7 days, the Tenant Manager Dashboard displays an alert message. You can go the Platform services endpoints page to see more details about the error.



One or more endpoints have experienced an error and might not be functioning properly. Go to the [Endpoints](#) page to view the error details. The last error occurred 2 hours ago.

The same error that appears on the Dashboard also appears at the top of the Platform services endpoints page. To view a more detailed error message:

Steps

1. From the list of endpoints, select the endpoint that has the error.
2. On the endpoint details page, select **Connection**. This tab displays only the most recent error for an endpoint and indicates how long ago the error occurred. Errors that include the red X icon  occurred within the past 7 days.

Overview

Display name:

my-endpoint-2

Type:

Search

URI:

http://10.96.104.30:9200

URN:

urn:sgws:es:::mydomain/sveloso/_doc

Connection

Configuration

Verify connection

Some errors might continue to appear after they are resolved. To see if an error is current or to force the removal of a resolved error, select **Test connection**.

Test connection

Last error details

2 hours ago

Endpoint failure: Endpoint has an AWS failure: RequestError: send request failed; caused by: url.Error; caused by: net.OpError; caused by: os.SyscallError (logID: 143H5UDUUKMGDRWJ)

Checking if an error is still current

Some errors might continue to be shown in the **Last error** column even after they are resolved. To see if an error is current or to force the removal of a resolved error from the table:

Steps

1. Select the endpoint.

The endpoint details page appears.

2. Select **Connection** > **Test connection**.

Selecting **Test connection** causes StorageGRID to validate that the platform services endpoint exists and that it can be reached with the current credentials. The connection to the endpoint is validated from one node at each site.

Resolving endpoint errors

You can use the **Last error** message on the endpoint details page to help determine what is causing the error. Some errors might require you to edit the endpoint to resolve the issue. For example, a CloudMirroring error

can occur if StorageGRID is unable to access the destination S3 bucket because it does not have the correct access permissions or the access key has expired. The message is “Either the endpoint credentials or the destination access needs to be updated,” and the details are “AccessDenied” or “InvalidAccessKeyld.”

If you need to edit the endpoint to resolve an error:, selecting **Test and save changes** causes StorageGRID to validate the updated endpoint and confirm that it can be reached with the current credentials. The connection to the endpoint is validated from one node at each site.

Steps

1. Select the endpoint.
2. On the endpoint details page, select **Configuration**.
3. Edit the endpoint configuration as needed.
4. Select **Connection > Test connection**.

Endpoint credentials with insufficient permissions

When StorageGRID validates a platform services endpoint, it confirms that the endpoint’s credentials can be used to contact the destination resource and it does a basic permissions check. However, StorageGRID does not validate all of the permissions required for certain platform services operations. For this reason, if you receive an error when attempting to use a platform service (such as “403 Forbidden”), check the permissions associated with the endpoint’s credentials.

Additional platform services troubleshooting

For additional information about troubleshooting platform services, see the instructions for administering StorageGRID.

[Administer StorageGRID](#)

Related information

[Creating a platform services endpoint](#)

[Testing the connection for a platform services endpoint](#)

[Editing a platform services endpoint](#)

Configuring CloudMirror replication

The CloudMirror replication service is one of the three StorageGRID platform services. You can use CloudMirror replication to automatically replicate objects to an external S3 bucket.

What you’ll need

- Platform services must be enabled for your tenant account by a StorageGRID administrator.
- You must have already created a bucket to act as the replication source.
- The endpoint that you intend to use as a destination for CloudMirror replication must already exist, and you must have its URN.
- You must belong to a user group that has the Manage All Buckets or the Root Access permission, which allows you to manage the settings for all S3 buckets in your tenant account. These permissions override the permission settings in group or bucket policies when configuring the bucket using the Tenant Manager.

About this task

CloudMirror replication copies objects from a source bucket to a destination bucket that is specified in an endpoint. To enable CloudMirror replication for a bucket, you must create and apply valid bucket replication configuration XML. The replication configuration XML must use the URN of an S3 bucket endpoint for each destination.



Replication is not supported for source or destination buckets with S3 Object Lock enabled.

For general information on bucket replication and how to configure it, see the Amazon documentation on cross-region replication (CRR). For information on how StorageGRID implements the S3 bucket replication configuration API, see the instructions for implementing S3 client applications.

If you enable CloudMirror replication on a bucket that contains objects, new objects added to the bucket are replicated, but the existing objects in the bucket are not. You must update existing objects to trigger replication.

If you specify a storage class in the replication configuration XML, StorageGRID uses that class when performing operations against the destination S3 endpoint. The destination endpoint must also support the specified storage class. Be sure to follow any recommendations provided by the destination system vendor.

Steps

1. Enable replication for your source bucket:

Use a text editor to create the replication configuration XML required to enable replication, as specified in the S3 replication API. When configuring the XML:

- Note that StorageGRID only supports V1 of the replication configuration. This means that StorageGRID does not support the use of the `Filter` element for rules, and follows V1 conventions for deletion of object versions. See the Amazon documentation on replication configuration for details.
- Use the URN of an S3 bucket endpoint as the destination.
- Optionally add the `<StorageClass>` element, and specify one of the following:
 - `STANDARD`: The default storage class. If you do not specify a storage class when you upload an object, the `STANDARD` storage class is used.
 - `STANDARD_IA`: (Standard - infrequent access.) Use this storage class for data that is accessed less frequently, but that still requires rapid access when needed.
 - `REDUCED_REDUNDANCY`: Use this storage class for noncritical, reproducible data that can be stored with less redundancy than the `STANDARD` storage class.
- If you specify a `Role` in the configuration XML it will be ignored. This value is not used by StorageGRID.

```
<ReplicationConfiguration>
  <Role></Role>
  <Rule>
    <Status>Enabled</Status>
    <Prefix>2020</Prefix>
    <Destination>
      <Bucket>urn:sgws:s3:::2017-records</Bucket>
      <StorageClass>STANDARD</StorageClass>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

2. In the Tenant Manager select **STORAGE (S3) > Buckets**.

3. Select the name of the source bucket.

The bucket details page appears.

4. Select **Platform services > Replication**.

5. Select the **Enable replication** check box.

6. Paste the replication configuration XML into the text box, and select **Save changes**.

For small objects, replication happens quickly.

Related information

[Understanding the CloudMirror replication service](#)

[Use S3](#)

[Creating a platform services endpoint](#)

Configuring event notifications

The notifications service is one of the three StorageGRID platform services. You can enable notifications for a bucket to send information about specified events to a destination service that supports the AWS Simple Notification Service™ (SNS).

What you'll need

- Platform services must be enabled for your tenant account by a StorageGRID administrator.
- You must have already created a bucket to act as the source of notifications.
- The endpoint that you intend to use as a destination for event notifications must already exist, and you must have its URN.
- You must belong to a user group that has the Manage All Buckets or the Root Access permission, which allows you to manage the settings for all S3 buckets in your tenant account. These permissions override the permission settings in group or bucket policies when configuring the bucket using the Tenant Manager.

About this task

After you configure event notifications, whenever a specified event occurs for an object in the source bucket, a notification is generated and sent to the Simple Notification Service (SNS) topic used as the destination endpoint. To enable notifications for a bucket, you must create and apply valid notification configuration XML. The notification configuration XML must use the URN of an event notifications endpoint for each destination.

For general information on event notifications and how to configure them, see Amazon documentation. For information on how StorageGRID implements the S3 bucket notification configuration API, see the instructions for implementing S3 client applications.

If you enable event notifications for a bucket that contains objects, notifications are sent only for actions that are performed after the notification configuration is saved.

Steps

1. Enable notifications for your source bucket:
 - Use a text editor to create the notification configuration XML required to enable event notifications, as specified in the S3 notification API.
 - When configuring the XML, use the URN of an event notifications endpoint as the destination topic.

```
<NotificationConfiguration>
  <TopicConfiguration>
    <Id>Image-created</Id>
    <Filter>
      <S3Key>
        <FilterRule>
          <Name>prefix</Name>
          <Value>images</Value>
        </FilterRule>
      </S3Key>
    </Filter>
    <Topic>arn:aws:sns:us-east-1:050340950352:sgws-topic</Topic>
    <Event>s3:ObjectCreated:*</Event>
  </TopicConfiguration>
</NotificationConfiguration>
```

2. In the Tenant Manager select **STORAGE (S3) > Buckets**.

3. Select the name of the source bucket.

The bucket details page appears.

4. Select **Platform services > Event notifications**.

5. Select the **Enable event notifications** check box.

6. Paste the notification configuration XML into the text box, and select **Save changes**.

Bucket options

Bucket access

Platform services

Replication

Disabled

▼

Event notifications

Disabled

▲

Enable the event notification service for an S3 bucket if you want StorageGRID to send notifications about specified events to a destination Amazon Simple Notification Service (SNS).

- Platform services must be enabled for your tenant account by a StorageGRID administrator.
- You must have already configured an endpoint for the destination of event notifications.
- You must specify the URN of that endpoint in the notification configuration XML for the source bucket.

☒ Enable event notifications

Clear

```

<NotificationConfiguration>
  <TopicConfiguration>
    <Id>Image-created</Id>
    <Filter>
      <S3Key>
        <FilterRule>
          <Name>prefix</Name>
          <Value>images/</Value>
        </FilterRule>
      </S3Key>
    </Filter>
    <Topic>arn:aws:sns:us-east-1:050340950352:sgws-topic</Topic>
  </TopicConfiguration>
</NotificationConfiguration>

```

Save changes



Platform services must be enabled for each tenant account by a StorageGRID administrator using the Grid Manager or Grid Management API. Contact your StorageGRID administrator if an error occurs when you save the configuration XML.

7. Verify that event notifications are configured correctly:

- Perform an action on an object in the source bucket that meets the requirements for triggering a notification as configured in the configuration XML.

In the example, an event notification is sent whenever an object is created with the `images/` prefix.

- b. Confirm that a notification has been delivered to the destination SNS topic.

For example, if your destination topic is hosted on the AWS Simple Notification Service (SNS), you could configure the service to send you an email when the notification is delivered.

```
{
  "Records": [
    {
      "eventVersion": "2.0",
      "eventSource": "sgws:s3",
      "eventTime": "2017-08-08T23:52:38Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "11111111111111111111"
      },
      "requestParameters": {
        "sourceIPAddress": "193.51.100.20"
      },
      "responseElements": {
        "x-amz-request-id": "122047343"
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "Image-created",
        "bucket": {
          "name": "test1",
          "ownerIdentity": {
            "principalId": "11111111111111111111"
          },
          "arn": "arn:sgws:s3:::test1"
        },
        "object": {
          "key": "images/cat.jpg",
          "size": 0,
          "eTag": "d41d8cd98f00b204e9800998ecf8427e",
          "sequencer": "14D90402421461C7"
        }
      }
    }
  ]
}
```

If the notification is received at the destination topic, you have successfully configured your source bucket for StorageGRID notifications.

Related information

[Understanding notifications for buckets](#)

[Use S3](#)

[Creating a platform services endpoint](#)

Using the search integration service

The search integration service is one of the three StorageGRID platform services. You can enable this service to send object metadata to a destination search index whenever an object is created, deleted, or its metadata or tags are updated.

You can configure search integration by using the Tenant Manager to apply custom StorageGRID configuration XML to a bucket.



Because the search integration service causes object metadata to be sent to a destination, its configuration XML is referred to as *metadata notification configuration XML*. This configuration XML is different than the *notification configuration XML* used to enable event notifications.

See the instructions for implementing S3 client applications for details about the following custom StorageGRID S3 REST API operations:

- DELETE Bucket metadata notification configuration request
- GET Bucket metadata notification configuration request
- PUT Bucket metadata notification configuration request

Related information

[Configuration XML for search integration](#)

[Object metadata included in metadata notifications](#)

[JSON generated by the search integration service](#)

[Configuring the search integration service](#)

[Use S3](#)

Configuration XML for search integration

The search integration service is configured using a set of rules contained within `<MetadataNotificationConfiguration>` and `</MetadataNotificationConfiguration>` tags. Each rule specifies the objects that the rule applies to, and the destination where StorageGRID should send those objects' metadata.

Objects can be filtered on the prefix of the object name. For example, you could send metadata for objects with the prefix `/images` to one destination, and metadata for objects with the prefix `/videos` to another. Configurations that have overlapping prefixes are not valid, and are rejected when they are submitted. For example, a configuration that includes one rule for objects with the prefix `test` and a second rule for objects with the prefix `test2` is not allowed.

Destinations must be specified using the URN of a StorageGRID endpoint that has been created for the search integration service. These endpoints refer to an index and type defined on an Elasticsearch cluster.

```
<MetadataNotificationConfiguration>
  <Rule>
    <ID>Rule-1</ID>
    <Status>rule-status</Status>
    <Prefix>key-prefix</Prefix>
    <Destination>
      <Urn>arn:aws:es:region:account-
ID:domain/mydomain/myindex/mytype</Urn>
    </Destination>
  </Rule>
  <Rule>
    <ID>Rule-2</ID>
    ...
  </Rule>
  ...
</MetadataNotificationConfiguration>
```

The table describes the elements in the metadata notification configuration XML.

Name	Description	Required
MetadataNotificationConfiguration	<p>Container tag for rules used to specify the objects and destination for metadata notifications.</p> <p>Contains one or more Rule elements.</p>	Yes
Rule	<p>Container tag for a rule that identifies the objects whose metadata should be added to a specified index.</p> <p>Rules with overlapping prefixes are rejected.</p> <p>Included in the MetadataNotificationConfiguration element.</p>	Yes
ID	<p>Unique identifier for the rule.</p> <p>Included in the Rule element.</p>	No
Status	<p>Status can be 'Enabled' or 'Disabled'. No action is taken for rules that are disabled.</p> <p>Included in the Rule element.</p>	Yes

Name	Description	Required
Prefix	<p>Objects that match the prefix are affected by the rule, and their metadata is sent to the specified destination.</p> <p>To match all objects, specify an empty prefix.</p> <p>Included in the Rule element.</p>	Yes
Destination	<p>Container tag for the destination of a rule.</p> <p>Included in the Rule element.</p>	Yes
Urn	<p>URN of the destination where object metadata is sent. Must be the URN of a StorageGRID endpoint with the following properties:</p> <ul style="list-style-type: none"> • <code>es</code> must be the third element. • The URN must end with the index and type where the metadata is stored, in the form <code>domain-name/myindex/mytype</code>. <p>Endpoints are configured using the Tenant Manager or Tenant Management API. They take the following form:</p> <ul style="list-style-type: none"> • <code>arn:aws:es:region:account-ID:domain/mydomain/myindex/mytype</code> • <code>urn:mysite:es:::mydomain/myindex/mytype</code> <p>The endpoint must be configured before the configuration XML is submitted, or configuration will fail with a 404 error.</p> <p>Urn is included in the Destination element.</p>	Yes

Use the sample metadata notification configuration XML to learn how to construct your own XML.

Metadata notification configuration that applies to all objects

In this example, object metadata for all objects is sent to the same destination.

```

<MetadataNotificationConfiguration>
  <Rule>
    <ID>Rule-1</ID>
    <Status>Enabled</Status>
    <Prefix></Prefix>
    <Destination>
      <Urn>urn:myes:es::sgws-notifications/test1/all</Urn>
    </Destination>
  </Rule>
</MetadataNotificationConfiguration>

```

Metadata notification configuration with two rules

In this example, object metadata for objects that match the prefix `/images` is sent to one destination, while object metadata for objects that match the prefix `/videos` is sent to a second destination.

```

<MetadataNotificationConfiguration>
  <Rule>
    <ID>Images-rule</ID>
    <Status>Enabled</Status>
    <Prefix>/images</Prefix>
    <Destination>
      <Urn>arn:aws:es:us-east-1:33333333:domain/es-
domain/graphics/imagetype</Urn>
    </Destination>
  </Rule>
  <Rule>
    <ID>Videos-rule</ID>
    <Status>Enabled</Status>
    <Prefix>/videos</Prefix>
    <Destination>
      <Urn>arn:aws:es:us-west-1:22222222:domain/es-
domain/graphics/videotype</Urn>
    </Destination>
  </Rule>
</MetadataNotificationConfiguration>

```

Related information

[Use S3](#)

[JSON generated by the search integration service](#)

[Configuring the search integration service](#)

Configuring the search integration service

The search integration service sends object metadata to a destination search index whenever an object is created, deleted, or its metadata or tags are updated.

What you'll need

- Platform services must be enabled for your tenant account by a StorageGRID administrator.
- You must have already created an S3 bucket whose contents you want to index.
- The endpoint that you intend to use as a destination for the search integration service must already exist, and you must have its URN.
- You must belong to a user group that has the Manage All Buckets or the Root Access permission, which allows you to manage the settings for all S3 buckets in your tenant account. These permissions override the permission settings in group or bucket policies when configuring the bucket using the Tenant Manager.

About this task

After you configure the search integration service for a source bucket, creating an object or updating an object's metadata or tags triggers object metadata to be sent to the destination endpoint. If you enable the search integration service for a bucket that already contains objects, metadata notifications are not automatically sent for existing objects. You must update these existing objects to ensure that their metadata is added to the destination search index.

Steps

1. Use a text editor to create the metadata notification XML required to enable search integration.
 - See the information about configuration XML for search integration.
 - When configuring the XML, use the URN of a search integration endpoint as the destination.

```
<MetadataNotificationConfiguration>
  <Rule>
    <Status>Enabled</Status>
    <Prefix></Prefix>
    <Destination>
      <Urn>arn:aws:es:us-east-
1:111111111111:domain/mydomain/myindex/mytype</Urn>
    </Destination>
  </Rule>
</MetadataNotificationConfiguration>
```

2. In the Tenant Manager select **STORAGE (S3) > Buckets**.
3. Select the name of the source bucket.

The bucket details page appears.

4. Select **Platform services > Search integration**
5. Select the **Enable search integration** check box.
6. Paste the metadata notification configuration into the text box, and select **Save changes**.

Bucket options

Bucket access

Platform services

Replication

Disabled

▼

Event notifications

Disabled

▼

Search integration

Disabled

▲

Enable the search integration service to send object metadata to a destination search index whenever an object is created, deleted, or its metadata or tags are updated.

- Platform services must be enabled for your tenant account by a StorageGRID administrator.
- You must have already configured an endpoint for the search integration service.
- You must specify the URN of that endpoint in the search integration configuration XML for the bucket you want to index.

☒ Enable search integration

Clear

```

<MetadataNotificationConfiguration>
  <Rule>
    <Status>Enabled</Status>
    <Prefix></Prefix>
    <Destination>
      <Urn>arn:aws:es:us-east-1:111111111111:domain/mydomain/myindex/mytype</Urn>
    </Destination>
  </Rule>
</MetadataNotificationConfiguration>

```

Save changes



Platform services must be enabled for each tenant account by a StorageGRID administrator using the Grid Manager or Management API. Contact your StorageGRID administrator if an error occurs when you save the configuration XML.

7. Verify that the search integration service is configured correctly:

- Add an object to the source bucket that meets the requirements for triggering a metadata notification as specified in the configuration XML.

In the example shown earlier, all objects added to the bucket trigger a metadata notification.

- Confirm that a JSON document that contains the object's metadata and tags was added to the search index specified in the endpoint.

After you finish

As necessary, you can disable search integration for a bucket using either of the following methods:

- Select **STORAGE (S3) > Buckets** and unselect the **Enable search integration** check box.
- If you are using the S3 API directly, use a DELETE Bucket metadata notification request. See the instructions for implementing S3 client applications.

Related information

[Understanding the search integration service](#)

[Configuration XML for search integration](#)

[Use S3](#)

[Creating a platform services endpoint](#)

JSON generated by the search integration service

When you enable the search integration service for a bucket, a JSON document is generated and sent to the destination endpoint each time object metadata or tags are added, updated, or deleted.

This example shows an example of the JSON that could be generated when an object with the key `SGWS/Tagging.txt` is created in a bucket named `test`. The `test` bucket is not versioned, so the `versionId` tag is empty.

```
{
  "bucket": "test",
  "key": "SGWS/Tagging.txt",
  "versionId": "",
  "accountId": "86928401983529626822",
  "size": 38,
  "md5": "3d6c7634a85436eee06d43415012855",
  "region": "us-east-1"
  "metadata": {
    "age": "25"
  },
  "tags": {
    "color": "yellow"
  }
}
```

Object metadata included in metadata notifications

The table lists all the fields that are included in the JSON document that is sent to the destination endpoint when search integration is enabled.

The document name includes the bucket name, object name, and version ID if present.

Type	Item name and description
Bucket and object information	bucket: Name of the bucket
	key: Object key name
	versionID: Object version, for objects in versioned buckets
	region: Bucket region, for example us-east-1
System metadata	size: Object size (in bytes) as visible to an HTTP client
	md5: Object hash
User metadata	metadata: All user metadata for the object, as key-value pairs key:value
Tags	tags: All object tags defined for the object, as key-value pairs key:value



For tags and user metadata, StorageGRID passes dates and numbers to Elasticsearch as strings or as S3 event notifications. To configure Elasticsearch to interpret these strings as dates or numbers, follow the Elasticsearch instructions for dynamic field mapping and for mapping date formats. You must enable the dynamic field mappings on the index before you configure the search integration service. After a document is indexed, you cannot edit the document's field types in the index.

Use S3

Learn how client applications can use the S3 API to interface with the StorageGRID system.

- [Support for the S3 REST API](#)
- [Configuring tenant accounts and connections](#)
- [How StorageGRID implements the S3 REST API](#)
- [S3 REST API supported operations and limitations](#)
- [StorageGRID S3 REST API operations](#)
- [Bucket and group access policies](#)
- [Configuring security for the REST API](#)
- [Monitoring and auditing operations](#)
- [Benefits of active, idle, and concurrent HTTP connections](#)

Support for the S3 REST API

StorageGRID supports the Simple Storage Service (S3) API, which is implemented as a set of Representational State Transfer (REST) web services. Support for the S3 REST API enables you to connect service-oriented applications developed for S3 web services with on-premises object storage that uses the StorageGRID system. This requires minimal changes to a client application's current use of S3 REST API calls.

- [Changes to S3 REST API support](#)
- [Supported versions](#)
- [Support for StorageGRID platform services](#)

Changes to S3 REST API support

You should be aware of changes to the StorageGRID system's support for the S3 REST API.

Release	Comments
11.5	<ul style="list-style-type: none">• Added support for managing bucket encryption.• Added support for S3 Object Lock and deprecated legacy Compliance requests.• Added support for using DELETE Multiple Objects on versioned buckets.• The <code>Content-MD5</code> request header is now correctly supported.
11.4	<ul style="list-style-type: none">• Added support for DELETE Bucket tagging, GET Bucket tagging, and PUT Bucket tagging. Cost allocation tags are not supported.• For buckets created in StorageGRID 11.4, restricting object key names to meet performance best practices is no longer required.• Added support for bucket notifications on the <code>s3:ObjectRestore:Post</code> event type.• AWS size limits for multipart parts are now enforced. Each part in a multipart upload must be between 5 MiB and 5 GiB. The last part can be smaller than 5 MiB.• Added support for TLS 1.3, and updated list of supported TLS cipher suites.• The CLB service is deprecated.

Release	Comments
11.3	<ul style="list-style-type: none"> • Added support for server-side encryption of object data with customer-provided keys (SSE-C). • Added support for DELETE, GET, and PUT Bucket lifecycle operations (Expiration action only) and for the <code>x-amz-expiration</code> response header. • Updated PUT Object, PUT Object - Copy, and Multipart Upload to describe the impact of ILM rules that use synchronous placement at ingest. • Updated list of supported TLS cipher suites. TLS 1.1 ciphers are no longer supported.
11.2	<p>Added support for POST Object restore for use with Cloud Storage Pools. Added support for using the AWS syntax for ARN, policy condition keys, and policy variables in group and bucket policies. Existing group and bucket policies that use the StorageGRID syntax will continue to be supported.</p> <p>Note: Uses of ARN/URN in other configuration JSON/XML, including those used in custom StorageGRID features, have not changed.</p>
11.1	Added support for Cross-Origin Resource Sharing (CORS), HTTP for S3 client connections to grid nodes, and compliance settings on buckets.
11.0	Added support for configuring platform services (CloudMirror replication, notifications, and Elasticsearch search integration) for buckets. Also added support for object tagging location constraints for buckets, and the Available consistency control setting.
10.4	Added support for ILM scanning changes to versioning, Endpoint Domain Names page updates, conditions and variables in policies, policy examples, and the PutOverwriteObject permission.
10.3	Added support for versioning.
10.2	Added support for group and bucket access policies, and for multipart copy (Upload Part - Copy).
10.1	Added support for multipart upload, virtual hosted-style requests, and v4 authentication.

Release	Comments
10.0	Initial support of the S3 REST API by the StorageGRID system. The currently supported version of the <i>Simple Storage Service API Reference</i> is 2006-03-01.

Supported versions

StorageGRID supports the following specific versions of S3 and HTTP.

Item	Version
S3 specification	<i>Simple Storage Service API Reference</i> 2006-03-01
HTTP	1.1 For more information about HTTP, see HTTP/1.1 (RFCs 7230-35). Note: StorageGRID does not support HTTP/1.1 pipelining.

Related information

[IETF RFC 2616: Hypertext Transfer Protocol \(HTTP/1.1\)](#)

[Amazon Web Services \(AWS\) Documentation: Amazon Simple Storage Service API Reference](#)

Support for StorageGRID platform services

StorageGRID platform services enable StorageGRID tenant accounts to leverage external services such as a remote S3 bucket, a Simple Notification Service (SNS) endpoint, or an Elasticsearch cluster to extend the services provided by a grid.

The following table summarizes the available platform services and the S3 APIs used to configure them.

Platform service	Purpose	S3 API used to configure the service
CloudMirror replication	Replicates objects from a source StorageGRID bucket to the configured remote S3 bucket.	PUT Bucket replication
Notifications	Sends notifications about events in a source StorageGRID bucket to a configured Simple Notification Service (SNS) endpoint.	PUT Bucket notification

Platform service	Purpose	S3 API used to configure the service
Search integration	Sends object metadata for objects stored in a StorageGRID bucket to a configured Elasticsearch index.	PUT Bucket metadata notification Note: This is a StorageGRID custom S3 API.

A grid administrator must enable the use of platform services for a tenant account before they can be used. Then, a tenant administrator must create an endpoint that represents the remote service in the tenant account. This step is required before a service can be configured.

Recommendations for using platform services

Before using platform services, you must be aware of the following recommendations:

- NetApp recommends that you allow no more than 100 active tenants with S3 requests requiring CloudMirror replication, notifications, and search integration. Having more than 100 active tenants can result in slower S3 client performance.
- If an S3 bucket in the StorageGRID system has both versioning and CloudMirror replication enabled, NetApp recommends that the destination endpoint also have S3 bucket versioning enabled. This allows CloudMirror replication to generate similar object versions on the endpoint.
- CloudMirror replication is not supported if the source bucket has S3 Object Lock enabled.
- CloudMirror replication will fail with an AccessDenied error if the destination bucket has legacy Compliance enabled.

Related information

[Use a tenant account](#)

[Administer StorageGRID](#)

[Operations on buckets](#)

[PUT Bucket metadata notification configuration request](#)

Configuring tenant accounts and connections

Configuring StorageGRID to accept connections from client applications requires creating one or more tenant accounts and setting up the connections.

Creating and configuring S3 tenant accounts

An S3 tenant account is required before S3 API clients can store and retrieve objects on StorageGRID. Each tenant account has its own account ID, groups and users, and containers and objects.

S3 tenant accounts are created by a StorageGRID grid administrator using the Grid Manager or the Grid Management API. When creating an S3 tenant account, the grid administrator specifies the following information:

- Display name for the tenant (the tenant's account ID is assigned automatically and cannot be changed).
- Whether the tenant account is allowed to use platform services. If the use of platform services is allowed,

the grid must be configured to support their use.

- Optionally, a storage quota for the tenant account—the maximum number of gigabytes, terabytes, or petabytes available for the tenant's objects. A tenant's storage quota represents a logical amount (object size), not a physical amount (size on disk).
- If identity federation is enabled for the StorageGRID system, which federated group has Root Access permission to configure the tenant account.
- If single sign-on (SSO) is not in use for the StorageGRID system, whether the tenant account will use its own identity source or share the grid's identity source, and the initial password for the tenant's local root user.

After an S3 tenant account is created, tenant users can access the Tenant Manager to perform tasks such as the following:

- Set up identity federation (unless the identity source is shared with the grid), and create local groups and users
- Manage S3 access keys
- Create and manage S3 buckets, including buckets that have S3 Object Lock enabled
- Use platform services (if enabled)
- Monitor storage usage



S3 tenant users can create and manage S3 buckets with the Tenant Manager, but they must have S3 access keys and use the S3 REST API to ingest and manage objects.

Related information

[Administer StorageGRID](#)

[Use a tenant account](#)

How client connections can be configured

A grid administrator makes configuration choices that affect how S3 clients connect to StorageGRID to store and retrieve data. The specific information you need to make a connection depends upon the configuration that was chosen.

Client applications can store or retrieve objects by connecting to any of the following:

- The Load Balancer service on Admin Nodes or Gateway Nodes, or optionally, the virtual IP address of a high availability (HA) group of Admin Nodes or Gateway Nodes
- The CLB service on Gateway Nodes, or optionally, the virtual IP address of a high availability group of Gateway Nodes



The CLB service is deprecated. Clients configured before the StorageGRID 11.3 release can continue to use the CLB service on Gateway Nodes. All other client applications that depend on StorageGRID to provide load balancing should connect using the Load Balancer service.

- Storage Nodes, with or without an external load balancer

When configuring StorageGRID, a grid administrator can use the Grid Manager or the Grid Management API to perform the following steps, all of which are optional:

1. Configure endpoints for the Load Balancer service.

You must configure endpoints to use the Load Balancer service. The Load Balancer service on Admin Nodes or Gateway Nodes distributes incoming network connections from client applications to Storage Nodes. When creating a load balancer endpoint, the StorageGRID administrator specifies a port number, whether the endpoint accepts HTTP or HTTPS connections, the type of client (S3 or Swift) that will use the endpoint, and the certificate to be used for HTTPS connections (if applicable).

2. Configure Untrusted Client Networks.

If a StorageGRID administrator configures a node's Client Network to be untrusted, the node only accepts inbound connections on the Client Network on ports that are explicitly configured as load balancer endpoints.

3. Configure high availability groups.

If an administrator creates an HA group, the network interfaces of multiple Admin Nodes or Gateway Nodes are placed into an active-backup configuration. Client connections are made using the virtual IP address of the HA group.

For more information about each option, see the instructions for administering StorageGRID.

Related information

[Administer StorageGRID](#)

Summary: IP addresses and ports for client connections

Client applications connect to StorageGRID using the IP address of a grid node and the port number of a service on that node. If high availability (HA) groups are configured, client applications can connect using the virtual IP address of the HA group.

Information required to make client connections

The table summarizes the different ways that clients can connect to StorageGRID and the IP addresses and ports that are used for each type of connection. Contact your StorageGRID administrator for more information, or see the instructions for administering StorageGRID for a description of how to find this information in the Grid Manager.

Where connection is made	Service that client connects to	IP address	Port
HA group	Load Balancer	Virtual IP address of an HA group	<ul style="list-style-type: none">• Load balancer endpoint port
HA group	CLB Note: The CLB service is deprecated.	Virtual IP address of an HA group	Default S3 ports: <ul style="list-style-type: none">• HTTPS: 8082• HTTP: 8084
Admin Node	Load Balancer	IP address of the Admin Node	<ul style="list-style-type: none">• Load balancer endpoint port

Where connection is made	Service that client connects to	IP address	Port
Gateway Node	Load Balancer	IP address of the Gateway Node	<ul style="list-style-type: none"> • Load balancer endpoint port
Gateway Node	CLB Note: The CLB service is deprecated.	IP address of the Gateway Node Note: By default, HTTP ports for CLB and LDR are not enabled.	Default S3 ports: <ul style="list-style-type: none"> • HTTPS: 8082 • HTTP: 8084
Storage Node	LDR	IP address of Storage Node	Default S3 ports: <ul style="list-style-type: none"> • HTTPS: 18082 • HTTP: 18084

Example

To connect an S3 client to the Load Balancer endpoint of an HA group of Gateway Nodes, use a URL structured as shown below:

- `https://VIP-of-HA-group:_LB-endpoint-port_`

For example, if the virtual IP address of the HA group is 192.0.2.5 and the port number of an S3 Load Balancer endpoint is 10443, then an S3 client could use the following URL to connect to StorageGRID:

- `https://192.0.2.5:10443`

It is possible to configure a DNS name for the IP address that clients use to connect to StorageGRID. Contact your local network administrator.

Related information

[Administer StorageGRID](#)

Deciding to use HTTPS or HTTP connections

When client connections are made using a Load Balancer endpoint, connections must be made using the protocol (HTTP or HTTPS) that was specified for that endpoint. To use HTTP for client connections to Storage Nodes or to the CLB service on Gateway Nodes, you must enable its use.

By default, when client applications connect to Storage Nodes or the CLB service on Gateway Nodes, they must use encrypted HTTPS for all connections. Optionally, you can enable less-secure HTTP connections by selecting the **Enable HTTP Connection** grid option in the Grid Manager. For example, a client application might use HTTP when testing the connection to a Storage Node in a non-production environment.



Be careful when enabling HTTP for a production grid since requests will be sent unencrypted.



The CLB service is deprecated.

If the **Enable HTTP Connection** option is selected, clients must use different ports for HTTP than they use for HTTPS. See the instructions for administering StorageGRID.

Related information

[Administer StorageGRID](#)

[Benefits of active, idle, and concurrent HTTP connections](#)

Endpoint domain names for S3 requests

Before you can use S3 domain names for client requests, a StorageGRID administrator must configure the system to accept connections that use S3 domain names in S3 path-style and S3 virtual hosted-style requests.

About this task

To enable you to use S3 virtual hosted style-requests, a grid administrator must perform the following tasks:

- Use the Grid Manager to add the S3 endpoint domain names to the StorageGRID system.
- Ensure that the certificate the client uses for HTTPS connections to StorageGRID is signed for all domain names that the client requires.

For example, if the endpoint is `s3.company.com`, the grid administrator must ensure that the certificate used for HTTPS connections includes the `s3.company.com` endpoint and the endpoint's wildcard Subject Alternative Name (SAN): `*.s3.company.com`.

- Configure the DNS server used by the client to include DNS records that match the endpoint domain names, including any required wildcard records.

If the client connects using the Load Balancer service, the certificate that the grid administrator configures is the certificate for the load balancer endpoint that the client uses.



Each load balancer endpoint has its own certificate, and each endpoint can be configured to recognize different endpoint domain names.

If the client connects Storage Nodes or to the CLB service on Gateway Nodes, the certificate that the grid administrator configures is the single custom server certificate used for the grid.



The CLB service is deprecated.

See the instructions for administering StorageGRID for more information.

After these steps have been completed, you can use virtual hosted-style requests (for example, `bucket.s3.company.com`).

Related information

[Administer StorageGRID](#)

[Configuring security for the REST API](#)

Testing your S3 REST API configuration

You can use the Amazon Web Services Command Line Interface (AWS CLI) to test your connection to the system and to verify that you can read and write objects to the system.

What you'll need

- You must have downloaded and installed the AWS CLI from aws.amazon.com/cli.
- You must have created an S3 tenant account in the StorageGRID system.

Steps

1. Configure the Amazon Web Services settings to use the account you created in the StorageGRID system:
 - a. Enter configuration mode: `aws configure`
 - b. Enter the AWS Access Key ID for the account you created.
 - c. Enter the AWS Secret Access key for the account you created.
 - d. Enter the default region to use, for example, `us-east-1`.
 - e. Enter the default output format to use, or press **Enter** to select JSON.

2. Create a bucket.

```
aws s3api --endpoint-url https://10.96.101.17:10443
--no-verify-ssl create-bucket --bucket testbucket
```

If the bucket is created successfully, the location of the bucket is returned, as seen in the following example:

```
"Location": "/testbucket"
```

3. Upload an object.

```
aws s3api --endpoint-url https://10.96.101.17:10443 --no-verify-ssl
put-object --bucket testbucket --key s3.pdf --body C:\s3-
test\upload\s3.pdf
```

If the object is uploaded successfully, an Etag is returned which is a hash of the object data.

4. List the contents of the bucket to verify that the object was uploaded.

```
aws s3api --endpoint-url https://10.96.101.17:10443 --no-verify-ssl
list-objects --bucket testbucket
```

5. Delete the object.

```
aws s3api --endpoint-url https://10.96.101.17:10443 --no-verify-ssl
delete-object --bucket testbucket --key s3.pdf
```

6. Delete the bucket.

```
aws s3api --endpoint-url https://10.96.101.17:10443 --no-verify-ssl
delete-bucket --bucket testbucket
```

How StorageGRID implements the S3 REST API

A client application can use S3 REST API calls to connect to StorageGRID to create, delete, and modify buckets, as well as storing and retrieving objects.

- [Conflicting client requests](#)
- [Consistency controls](#)
- [How StorageGRID ILM rules manage objects](#)
- [Object versioning](#)
- [Recommendations for implementing the S3 REST API](#)

Conflicting client requests

Conflicting client requests, such as two clients writing to the same key, are resolved on a “latest-wins” basis.

The timing for the “latest-wins” evaluation is based on when the StorageGRID system completes a given request, and not on when S3 clients begin an operation.

Consistency controls

Consistency controls provide a trade-off between the availability of the objects and the consistency of those objects across different Storage Nodes and sites, as required by your application.

By default, StorageGRID guarantees read-after-write consistency for newly created objects. Any GET following a successfully completed PUT will be able to read the newly written data. Overwrites of existing objects, metadata updates, and deletes are eventually consistent. Overwrites generally take seconds or minutes to propagate, but can take up to 15 days.

If you want to perform object operations at a different consistency level, you can specify a consistency control for each bucket or for each API operation.

Consistency controls

The consistency control affects how the metadata that StorageGRID uses to track objects is distributed between nodes, and therefore the availability of objects for client requests.

You can set the consistency control for a bucket or an API operation to one of the following values:

Consistency control	Description
all	All nodes receive the data immediately, or the request will fail.

Consistency control	Description
strong-global	Guarantees read-after-write consistency for all client requests across all sites.
strong-site	Guarantees read-after-write consistency for all client requests within a site.
read-after-new-write	<p>(Default) Provides read-after-write consistency for new objects and eventual consistency for object updates. Offers high availability and data protection guarantees. Matches Amazon S3 consistency guarantees.</p> <p>Note: If your application uses HEAD requests on objects that do not exist, you might receive a high number of 500 Internal Server errors if one or more Storage Nodes are unavailable. To prevent these errors, set the consistency control to “available” unless you require consistency guarantees similar to Amazon S3.</p>
available (eventual consistency for HEAD operations)	Behaves the same as the “read-after-new-write” consistency level, but only provides eventual consistency for HEAD operations. Offers higher availability for HEAD operations than “read-after-new-write” if Storage Nodes are unavailable. Differs from Amazon S3 consistency guarantees for HEAD operations only.

Using the “read-after-new-write” and “available” consistency controls

When a HEAD or GET operation uses the “read-after-new-write” consistency control or a GET operation uses the “available” consistency control, StorageGRID performs the lookup in multiple steps, as follows:

- It first looks up the object using a low consistency.
- If that lookup fails, it repeats the lookup at the next consistency level until it reaches the highest consistency level, “all,” which requires all copies of the object metadata to be available.

If a HEAD or GET operation uses the “read-after-new-write” consistency control but the object does not exist, the object lookup will always reach the “all” consistency level. Because this consistency level requires all copies of the object metadata to be available, you can receive a high number of 500 Internal Server errors if one or more Storage Nodes are unavailable.

Unless you require consistency guarantees similar to Amazon S3, you can prevent these errors for HEAD operations by setting the consistency control to “available.” When a HEAD operation uses the “available” consistency control, StorageGRID provides eventual consistency only. It does not retry a failed operation until it reaches the “all” consistency level, so it does not require that all copies of the object metadata be available.

Specifying the consistency control for an API operation

To set the consistency control for an individual API operation, consistency controls must be supported for the

operation, and you must specify the consistency control in the request header. This example sets the consistency control to “strong-site” for a GET Object operation.

```
GET /bucket/object HTTP/1.1
Date: date
Authorization: <em>authorization name</em>
Host: <em>host</em>
Consistency-Control: strong-site
```



You must use the same consistency control for both the PUT Object and GET Object operations.

Specifying the consistency control for a bucket

To set the consistency control for bucket, you can use the StorageGRID PUT Bucket consistency request and the GET Bucket consistency request. Or you can use the Tenant Manager or the Tenant Management API.

When setting the consistency controls for a bucket, be aware of the following:

- Setting the consistency control for a bucket determines which consistency control is used for S3 operations performed on the objects in the bucket or on the bucket configuration. It does not affect operations on the bucket itself.
- The consistency control for an individual API operation overrides the consistency control for the bucket.
- In general, buckets should use the default consistency control, “read-after-new-write.” If requests are not working correctly, change the application client behavior if possible. Or, configure the client to specify the consistency control for each API request. Set the consistency control at the bucket level only as a last resort.

How consistency controls and ILM rules interact to affect data protection

Both your choice of consistency control and your ILM rule affect how objects are protected. These settings can interact.

For example, the consistency control used when an object is stored affects the initial placement of object metadata, while the ingest behavior selected for the ILM rule affects the initial placement of object copies. Because StorageGRID requires access to both an object’s metadata and its data to fulfill client requests, selecting matching levels of protection for the consistency level and ingest behavior can provide better initial data protection and more predictable system responses.

The following ingest behaviors are available for ILM rules:

- **Strict:** All copies specified in the ILM rule must be made before success is returned to the client.
- **Balanced:** StorageGRID attempts to make all copies specified in the ILM rule at ingest; if this is not possible, interim copies are made and success is returned to the client. The copies specified in the ILM rule are made when possible.
- **Dual Commit:** StorageGRID immediately makes interim copies of the object and returns success to the client. Copies specified in the ILM rule are made when possible.



Before selecting the ingest behavior for an ILM rule, read the full description of these settings in the instructions for managing objects with information lifecycle management.

Example of how the consistency control and ILM rule can interact

Suppose you have a two-site grid with the following ILM rule and the following consistency level setting:

- **ILM rule:** Create two object copies, one at the local site and one at a remote site. The Strict ingest behavior is selected.
- **Consistency level:** “strong-global” (Object metadata is immediately distributed to all sites.)

When a client stores an object to the grid, StorageGRID makes both object copies and distributes metadata to both sites before returning success to the client.

The object is fully protected against loss at the time of the ingest successful message. For example, if the local site is lost shortly after ingest, copies of both the object data and the object metadata still exist at the remote site. The object is fully retrievable.

If you instead used the same ILM rule and the “strong-site” consistency level, the client might receive a success message after object data is replicated to the remote site but before object metadata is distributed there. In this case, the level of protection of object metadata does not match the level of protection for object data. If the local site is lost shortly after ingest, object metadata is lost. The object cannot be retrieved.

The inter-relationship between consistency levels and ILM rules can be complex. Contact NetApp if you require assistance.

Related information

[Manage objects with ILM](#)

[GET Bucket consistency request](#)

[PUT Bucket consistency request](#)

How StorageGRID ILM rules manage objects

The grid administrator creates information lifecycle management (ILM) rules to manage object data ingested into the StorageGRID system from S3 REST API client applications. These rules are then added to the ILM policy to determine how and where object data is stored over time.

ILM settings determine the following aspects of an object:

- **Geography**

The location of an object's data, either within the StorageGRID system (storage pool) or in a Cloud Storage Pool.

- **Storage grade**

The type of storage used to store object data: for example flash or spinning disk.

- **Loss protection**

How many copies are made and the types of copies that are created: replication, erasure coding, or both.

- **Retention**

The changes over time to how an object's data is managed, where it is stored, and how it is protected from loss.

- **Protection during ingest**

The method used to protect object data during ingest: synchronous placement (using the Balanced or Strict options for Ingest Behavior), or making interim copies (using the Dual commit option).

ILM rules can filter and select objects. For objects ingested using S3, ILM rules can filter objects based on the following metadata:

- Tenant Account
- Bucket Name
- Ingest Time
- Key
- Last Access Time



By default, updates to last access time are disabled for all S3 buckets. If your StorageGRID system includes an ILM rule that uses the Last Access Time option, you must enable updates to last access time for the S3 buckets specified in that rule. You can enable last access time updates using the PUT Bucket last access time request, the **S3 > Buckets > Configure Last Access Time** check box in the Tenant Manager, or using the Tenant Management API. When enabling last access time updates, be aware that StorageGRID performance might be reduced, especially in systems with small objects.

- Location Constraint
- Object Size
- User Metadata
- Object Tag

For more information about ILM, see the instructions for managing objects with information lifecycle management.

Related information

[Use a tenant account](#)

[Manage objects with ILM](#)

[PUT Bucket last access time request](#)

Object versioning

You can use versioning to retain multiple versions of an object, which protects against accidental deletion of objects, and enables you to retrieve and restore earlier versions of an object.

The StorageGRID system implements versioning with support for most features, and with some limitations. StorageGRID supports up to 1,000 versions of each object.

Object versioning can be combined with StorageGRID information lifecycle management (ILM) or with S3

bucket lifecycle configuration. You must explicitly enable versioning for each bucket to turn on this functionality for the bucket. Each object in your bucket is assigned a version ID, which is generated by the StorageGRID system.

Using MFA (multi-factor authentication) Delete is not supported.



Versioning can be enabled only on buckets created with StorageGRID version 10.3 or later.

ILM and versioning

ILM policies are applied to each version of an object. An ILM scanning process continuously scans all objects and re-evaluates them against the current ILM policy. Any changes you make to ILM policies are applied to all previously ingested objects. This includes previously ingested versions if versioning is enabled. ILM scanning applies new ILM changes to previously ingested objects.

For S3 objects in versioning-enabled buckets, versioning support allows you to create ILM rules that use Noncurrent Time as the Reference Time. When an object is updated, its previous versions become noncurrent. Using a noncurrent time filter allows you to create policies that reduce the storage impact of previous versions of objects.



When you upload a new version of an object using a multipart upload operation, the noncurrent time for the original version of the object reflects when the multipart upload was created for the new version, not when the multipart upload was completed. In limited cases, the noncurrent time for the original version might be hours or days earlier than the time for the current version.

See the instructions for managing objects with information lifecycle management for an example ILM policy for S3 versioned objects.

Related information

[Manage objects with ILM](#)

Recommendations for implementing the S3 REST API

You should follow these recommendations when implementing the S3 REST API for use with StorageGRID.

Recommendations for HEADs to non-existent objects

If your application routinely checks to see if an object exists at a path where you do not expect the object to actually exist, you should use the “Available” consistency control. For example, you should use the “Available” consistency control if your application HEADs a location before PUT-ing to it.

Otherwise, if the HEAD operation does not find the object, you might receive a high number of 500 Internal Server errors if one or more Storage Nodes are unavailable.

You can set the “Available” consistency control for each bucket using the PUT Bucket consistency request, or you can specify the consistency control in the request header for an individual API operation.

Recommendations for object keys

For buckets that are created in StorageGRID 11.4 or later, restricting object key names to meet performance best practices is no longer required. For example, you can now use random values for the first four characters of object key names.

For buckets that were created in releases earlier than StorageGRID 11.4, continue to follow these recommendations for object key names:

- You should not use random values as the first four characters of object keys. This is in contrast to the former AWS recommendation for key prefixes. Instead, you should use non-random, non-unique prefixes, such as `image`.
- If you do follow the former AWS recommendation to use random and unique characters in key prefixes, you should prefix the object keys with a directory name. That is, use this format:

```
mybucket/mydir/f8e3-image3132.jpg
```

Instead of this format:

```
mybucket/f8e3-image3132.jpg
```

Recommendations for “range reads”

If the **Compress Stored Objects** option is selected (**Configuration > Grid Options**), S3 client applications should avoid performing GET Object operations that specify a range of bytes be returned. These “range read” operations are inefficient because StorageGRID must effectively uncompress the objects to access the requested bytes. GET Object operations that request a small range of bytes from a very large object are especially inefficient; for example, it is very inefficient to read a 10 MB range from a 50 GB compressed object.

If ranges are read from compressed objects, client requests can time out.



If you need to compress objects and your client application must use range reads, increase the read timeout for the application.

Related information

[Consistency controls](#)

[PUT Bucket consistency request](#)

[Administer StorageGRID](#)

S3 REST API supported operations and limitations

The StorageGRID system implements the Simple Storage Service API (API Version 2006-03-01) with support for most operations, and with some limitations. You need to understand the implementation details when you are integrating S3 REST API client applications.

The StorageGRID system supports both virtual hosted-style requests and path-style requests.

- [Authenticating requests](#)
- [Operations on the service](#)
- [Operations on buckets](#)

- [Custom operations on buckets](#)
- [Operations on objects](#)
- [Operations for multipart uploads](#)
- [Error responses](#)

Date handling

The StorageGRID implementation of the S3 REST API only supports valid HTTP date formats.

The StorageGRID system only supports valid HTTP date formats for any headers that accept date values. The time portion of the date can be specified in Greenwich Mean Time (GMT) format, or in Universal Coordinated Time (UTC) format with no time zone offset (+0000 must be specified). If you include the `x-amz-date` header in your request, it overrides any value specified in the Date request header. When using AWS Signature Version 4, the `x-amz-date` header must be present in the signed request because the date header is not supported.

Common request headers

The StorageGRID system supports common request headers defined by the *Simple Storage Service API Reference*, with one exception.

Request header	Implementation
Authorization	<p>Full support for AWS Signature Version 2</p> <p>Support for AWS Signature Version 4, with the following exceptions:</p> <ul style="list-style-type: none"> • The SHA256 value is not calculated for the body of the request. The user-submitted value is accepted without validation, as if the value <code>UNSIGNED-PAYLOAD</code> had been provided for the <code>x-amz-content-sha256</code> header.
x-amz-security-token	Not implemented. Returns <code>XNotImplemented</code> .

Common response headers

The StorageGRID system supports all of the common response headers defined by the *Simple Storage Service API Reference*, with one exception.

Response header	Implementation
x-amz-id-2	Not used

Related information

[Amazon Web Services \(AWS\) Documentation: Amazon Simple Storage Service API Reference](#)

Authenticating requests

The StorageGRID system supports both authenticated and anonymous access to objects using the S3 API.

The S3 API supports Signature version 2 and Signature version 4 for authenticating S3 API requests.

Authenticated requests must be signed using your access key ID and secret access key.

The StorageGRID system supports two authentication methods: the HTTP `Authorization` header and using query parameters.

Using the HTTP Authorization header

The HTTP `Authorization` header is used by all S3 API operations except Anonymous requests where permitted by the bucket policy. The `Authorization` header contains all of the required signing information to authenticate a request.

Using query parameters

You can use query parameters to add authentication information to a URL. This is known as presigning the URL, which can be used to grant temporary access to specific resources. Users with the presigned URL do not need to know the secret access key in order to access the resource, which enables you to provide third-party restricted access to a resource.

Operations on the service

The StorageGRID system supports the following operations on the service.

Operation	Implementation
GET Service	Implemented with all Amazon S3 REST API behavior.
GET Storage Usage	The GET Storage Usage request tells you the total amount of storage in use by an account, and for each bucket associated with the account. This is an operation on the service with a path of <code>/</code> and a custom query parameter (<code>?x-ntap-sg-usage</code>) added.
OPTIONS /	Client applications can issue <code>OPTIONS /</code> requests to the S3 port on a Storage Node, without providing S3 authentication credentials, to determine whether the Storage Node is available. You can use this request for monitoring, or to allow external load balancers to identify when a Storage Node is down.

Related information

[GET Storage Usage request](#)

Operations on buckets

The StorageGRID system supports a maximum of 1,000 buckets for each S3 tenant

account.

Bucket name restrictions follow the AWS US Standard region restrictions, but you should further restrict them to DNS naming conventions in order to support S3 virtual hosted-style requests.

[Amazon Web Services \(AWS\) Documentation: Bucket Restrictions and Limitations](#)

[Endpoint domain names for S3 request](#)

The GET Bucket (List Objects) and GET Bucket versions operations support StorageGRID consistency controls.

You can check whether updates to last access time are enabled or disabled for individual buckets.

The following table describes how StorageGRID implements S3 REST API bucket operations. To perform any of these operations, the necessary access credentials must be provided for the account.

Operation	Implementation
DELETE Bucket	Implemented with all Amazon S3 REST API behavior.
DELETE Bucket cors	This operation deletes the CORS configuration for the bucket.
DELETE Bucket encryption	This operation deletes the default encryption from the bucket. Existing encrypted objects remain encrypted, but any new objects added to the bucket are not encrypted.
DELETE Bucket lifecycle	This operation deletes the lifecycle configuration from the bucket.
DELETE Bucket policy	This operation deletes the policy attached to the bucket.
DELETE Bucket replication	This operation deletes the replication configuration attached to the bucket.
DELETE Bucket tagging	This operation uses the <code>tagging</code> subresource to remove all tags from a bucket.

Operation	Implementation
GET Bucket (List Objects), version 1 and version 2	<p>This operation returns some or all (up to 1,000) of the objects in a bucket. The Storage Class for objects can have either of two values, even if the object was ingested with the <code>REDUCED_REDUNDANCY</code> storage class option:</p> <ul style="list-style-type: none"> • <code>STANDARD</code>, which indicates the object is stored in a storage pool consisting of Storage Nodes. • <code>GLACIER</code>, which indicates that the object has been moved to the external bucket specified by the Cloud Storage Pool. <p>If the bucket contains large numbers of deleted keys that have the same prefix, the response might include some <code>CommonPrefixes</code> that do not contain keys.</p>
GET Bucket acl	This operation returns a positive response and the ID, DisplayName, and Permission of the bucket owner, indicating that the owner has full access to the bucket.
GET Bucket cors	This operation returns the <code>cors</code> configuration for the bucket.
GET Bucket encryption	This operation returns the default encryption configuration for the bucket.
GET Bucket lifecycle	This operation returns the lifecycle configuration for the bucket.
GET Bucket location	This operation returns the region that was set using the <code>LocationConstraint</code> element in the <code>PUT Bucket</code> request. If the bucket's region is <code>us-east-1</code> , an empty string is returned for the region.
GET Bucket notification	This operation returns the notification configuration attached to the bucket.
GET Bucket Object versions	With <code>READ</code> access on a bucket, this operation with the <code>versions</code> subresource lists metadata of all of the versions of objects in the bucket.
GET Bucket policy	This operation returns the policy attached to the bucket.
GET Bucket replication	This operation returns the replication configuration attached to the bucket.

Operation	Implementation
GET Bucket tagging	This operation uses the <code>tagging</code> subresource to return all tags for a bucket.
GET Bucket versioning	This implementation uses the <code>versioning</code> subresource to return the versioning state of a bucket. The versioning state returned indicates if the bucket is “Unversioned” or if the bucket is version “Enabled” or “Suspended.”
GET Object Lock Configuration	This operation determines if S3 Object Lock is enabled for a bucket. Using S3 Object Lock
HEAD Bucket	This operation determines if a bucket exists and you have permission to access it.

Operation	Implementation
PUT Bucket	<p>This operation creates a new bucket. By creating the bucket, you become the bucket owner.</p> <ul style="list-style-type: none"> • Bucket names must comply with the following rules: <ul style="list-style-type: none"> ◦ Must be unique across each StorageGRID system (not just unique within the tenant account). ◦ Must be DNS compliant. ◦ Must contain at least 3 and no more than 63 characters. ◦ Can be a series of one or more labels, with adjacent labels separated by a period. Each label must start and end with a lowercase letter or a number and can only use lowercase letters, numbers, and hyphens. ◦ Must not look like a text-formatted IP address. ◦ Should not use periods in virtual hosted style requests. Periods will cause problems with server wildcard certificate verification. • By default, buckets are created in the <code>us-east-1</code> region; however, you can use the <code>LocationConstraint</code> request element in the request body to specify a different region. When using the <code>LocationConstraint</code> element, you must specify the exact name of a region that has been defined using the Grid Manager or the Grid Management API. Contact your system administrator if you do not know the region name you should use. Note: An error will occur if your PUT Bucket request uses a region that has not been defined in StorageGRID. • You can include the <code>x-amz-bucket-object-lock-enabled</code> request header to create a bucket with S3 Object Lock enabled. <p>You must enable S3 Object Lock when you create the bucket. You cannot add or disable S3 Object Lock after a bucket is created. S3 Object Lock requires bucket versioning, which is enabled automatically when you create the bucket.</p> <p>Using S3 Object Lock</p>

Operation	Implementation
PUT Bucket cors	<p>This operation sets the CORS configuration for a bucket so that the bucket can service cross-origin requests. Cross-origin resource sharing (CORS) is a security mechanism that allows client web applications in one domain to access resources in a different domain. For example, suppose you use an S3 bucket named <code>images</code> to store graphics. By setting the CORS configuration for the <code>images</code> bucket, you can allow the images in that bucket to be displayed on the website <code>http://www.example.com</code>.</p>
PUT Bucket encryption	<p>This operation sets the default encryption state of an existing bucket. When bucket-level encryption is enabled, any new objects added to the bucket are encrypted. StorageGRID supports server-side encryption with StorageGRID-managed keys. When specifying the server-side encryption configuration rule, set the <code>SSEAlgorithm</code> parameter to <code>AES256</code>, and do not use the <code>KMSMasterKeyID</code> parameter.</p> <p>Bucket default encryption configuration is ignored if the object upload request already specifies encryption (that is, if the request includes the <code>x-amz-server-side-encryption-*</code> request header).</p>

Operation	Implementation
PUT Bucket lifecycle	<p>This operation creates a new lifecycle configuration for the bucket or replaces an existing lifecycle configuration. StorageGRID supports up to 1,000 lifecycle rules in a lifecycle configuration. Each rule can include the following XML elements:</p> <ul style="list-style-type: none"> • Expiration (Days, Date) • NoncurrentVersionExpiration (NoncurrentDays) • Filter (Prefix, Tag) • Status • ID <p>StorageGRID does not support these actions:</p> <ul style="list-style-type: none"> • AbortIncompleteMultipartUpload • ExpiredObjectDeleteMarker • Transition <p>To understand how the Expiration action in a bucket lifecycle interacts with ILM placement instructions, see “How ILM operates throughout an object’s life” in the instructions for managing objects with information lifecycle management.</p> <p>Note: Bucket lifecycle configuration can be used with buckets that have S3 Object Lock enabled, but bucket lifecycle configuration is not supported for legacy Compliant buckets.</p>

Operation	Implementation
PUT Bucket notification	<p>This operation configures notifications for the bucket using the notification configuration XML included in the request body. You should be aware of the following implementation details:</p> <ul style="list-style-type: none"> StorageGRID supports Simple Notification Service (SNS) topics as destinations. Simple Queue Service (SQS) or Amazon Lambda endpoints are not supported. The destination for notifications must be specified as the URN of an StorageGRID endpoint. Endpoints can be created using the Tenant Manager or the Tenant Management API. <p>The endpoint must exist for notification configuration to succeed. If the endpoint does not exist, a 400 Bad Request error is returned with the code <code>InvalidArgument</code>.</p> <ul style="list-style-type: none"> You cannot configure a notification for the following event types. These event types are not supported. <ul style="list-style-type: none"> <code>s3:ReducedRedundancyLostObject</code> <code>s3:ObjectRestore:Completed</code> Event notifications sent from StorageGRID use the standard JSON format except that they do not include some keys and use specific values for others, as shown in the following listing: eventSource <pre>sgws:s3</pre> awsRegion <pre>not included</pre> x-amz-id-2 <pre>not included</pre> arn <pre>urn:sgws:s3:::bucket_name</pre>
PUT Bucket policy	This operation sets the policy attached to the bucket.

Operation	Implementation
PUT Bucket replication	<p>This operation configures StorageGRID CloudMirror replication for the bucket using the replication configuration XML provided in the request body. For CloudMirror replication, you should be aware of the following implementation details:</p> <ul style="list-style-type: none"> • StorageGRID only supports V1 of the replication configuration. This means that StorageGRID does not support the use of the <code>Filter</code> element for rules, and follows V1 conventions for deletion of object versions. See the Amazon documentation on replication configuration for details. • Bucket replication can be configured on versioned or unversioned buckets. • You can specify a different destination bucket in each rule of the replication configuration XML. A source bucket can replicate to more than one destination bucket. • Destination buckets must be specified as the URN of StorageGRID endpoints as specified in the Tenant Manager or the Tenant Management API. <p>The endpoint must exist for replication configuration to succeed. If the endpoint does not exist, the request fails as a 400 Bad Request. The error message states: Unable to save the replication policy. The specified endpoint URN does not exist: <i>URN</i>.</p> <ul style="list-style-type: none"> • You do not need to specify a <code>Role</code> in the configuration XML. This value is not used by StorageGRID and will be ignored if submitted. • If you omit the storage class from the configuration XML, StorageGRID uses the <code>STANDARD</code> storage class by default. • If you delete an object from the source bucket or you delete the source bucket itself, the cross-region replication behavior is as follows: <ul style="list-style-type: none"> ◦ If you delete the object or bucket before it has been replicated, the object/bucket is not replicated and you are not notified. ◦ If you delete the object or bucket after it has been replicated, StorageGRID follows standard Amazon S3 delete behavior for V1 of cross-region replication.

Operation	Implementation
PUT Bucket tagging	<p>This operation uses the <code>tagging</code> subresource to add or update a set of tags for a bucket. When adding bucket tags, be aware of the following limitations:</p> <ul style="list-style-type: none"> • Both StorageGRID and Amazon S3 support up to 50 tags for each bucket. • Tags associated with a bucket must have unique tag keys. A tag key can be up to 128 Unicode characters in length. • Tag values can be up to 256 Unicode characters in length. • Key and values are case sensitive.
PUT Bucket versioning	<p>This implementation uses the <code>versioning</code> subresource to set the versioning state of an existing bucket. You can set the versioning state with one of the following values:</p> <ul style="list-style-type: none"> • Enabled: Enables versioning for the objects in the bucket. All objects added to the bucket receive a unique version ID. • Suspended: Disables versioning for the objects in the bucket. All objects added to the bucket receive the version ID <code>null</code>.

Related information

[Amazon Web Services \(AWS\) Documentation: Cross-Region Replication](#)

[Consistency controls](#)

[GET Bucket last access time request](#)

[Bucket and group access policies](#)

[Using S3 Object Lock](#)

[S3 operations tracked in the audit logs](#)

[Manage objects with ILM](#)

[Use a tenant account](#)

Creating an S3 lifecycle configuration

You can create an S3 lifecycle configuration to control when specific objects are deleted from the StorageGRID system.

The simple example in this section illustrates how an S3 lifecycle configuration can control when certain objects are deleted (expired) from specific S3 buckets. The example in this section is for illustration purposes

only. For complete details on creating S3 lifecycle configurations, see the section on object lifecycle management in the *Amazon Simple Storage Service Developer Guide*. Note that StorageGRID only supports Expiration actions; it does not support Transition actions.

[Amazon Simple Storage Service Developer Guide: Object lifecycle management](#)

What a lifecycle configuration is

A lifecycle configuration is a set of rules that are applied to the objects in specific S3 buckets. Each rule specifies which objects are affected and when those objects will expire (on a specific date or after some number of days).

StorageGRID supports up to 1,000 lifecycle rules in a lifecycle configuration. Each rule can include the following XML elements:

- Expiration: Delete an object when a specified date is reached or when a specified number of days is reached, starting from when the object was ingested.
- NoncurrentVersionExpiration: Delete an object when a specified number of days is reached, starting from when the object became noncurrent.
- Filter (Prefix, Tag)
- Status
- ID

If you apply a lifecycle configuration to a bucket, the lifecycle settings for the bucket always override StorageGRID ILM settings. StorageGRID uses the Expiration settings for the bucket, not ILM, to determine whether to delete or retain specific objects.

As a result, an object might be removed from the grid even though the placement instructions in an ILM rule still apply to the object. Or, an object might be retained on the grid even after any ILM placement instructions for the object have lapsed. For details, see “How ILM operates throughout an object’s life” in the instructions for managing objects with information lifecycle management.



Bucket lifecycle configuration can be used with buckets that have S3 Object Lock enabled, but bucket lifecycle configuration is not supported for legacy Compliant buckets.

StorageGRID supports the use of the following bucket operations to manage lifecycle configurations:

- DELETE Bucket lifecycle
- GET Bucket lifecycle
- PUT Bucket lifecycle

Creating the lifecycle configuration

As the first step in creating a lifecycle configuration, you create a JSON file that includes one or more rules. For example, this JSON file includes three rules, as follows:

1. Rule 1 applies only to objects that match the prefix `category1/` and that have a `key2` value of `tag2`. The `Expiration` parameter specifies that objects matching the filter will expire at midnight on 22 August 2020.
2. Rule 2 applies only to objects that match the prefix `category2/`. The `Expiration` parameter specifies that objects matching the filter will expire 100 days after they are ingested.



Rules that specify a number of days are relative to when the object was ingested. If the current date exceeds the ingest date plus the number of days, some objects might be removed from the bucket as soon as the lifecycle configuration is applied.

3. Rule 3 applies only to objects that match the prefix `category3/`. The `Expiration` parameter specifies that any noncurrent versions of matching objects will expire 50 days after they become noncurrent.

```

{
  "Rules": [
    {
      "ID": "rule1",
      "Filter": {
        "And": {
          "Prefix": "category1/",
          "Tags": [
            {
              "Key": "key2",
              "Value": "tag2"
            }
          ]
        }
      },
      "Expiration": {
        "Date": "2020-08-22T00:00:00Z"
      },
      "Status": "Enabled"
    },
    {
      "ID": "rule2",
      "Filter": {
        "Prefix": "category2/"
      },
      "Expiration": {
        "Days": 100
      },
      "Status": "Enabled"
    },
    {
      "ID": "rule3",
      "Filter": {
        "Prefix": "category3/"
      },
      "NoncurrentVersionExpiration": {
        "NoncurrentDays": 50
      },
      "Status": "Enabled"
    }
  ]
}

```

Applying a lifecycle configuration to a bucket

After you have created the lifecycle configuration file, you apply it to a bucket by issuing a PUT Bucket lifecycle request.

This request applies the lifecycle configuration in the example file to objects in a bucket named `testbucket:bucket`

```
aws s3api --endpoint-url <StorageGRID endpoint> put-bucket-lifecycle-configuration
--bucket testbucket --lifecycle-configuration file://bktjson.json
```

To validate that a lifecycle configuration was successfully applied to the bucket, issue a GET Bucket lifecycle request. For example:

```
aws s3api --endpoint-url <StorageGRID endpoint> get-bucket-lifecycle-configuration
--bucket testbucket
```

A successful response lists the lifecycle configuration you just applied.

Validating that bucket lifecycle expiration applies to an object

You can determine if an expiration rule in the lifecycle configuration applies to a specific object when issuing a PUT Object, HEAD Object, or GET Object request. If a rule applies, the response includes an `Expiration` parameter that indicates when the object expires and which expiration rule was matched.



Because bucket lifecycle overrides ILM, the `expiry-date` shown is the actual date the object will be deleted. For details, see “How object retention is determined” in the instructions for performing StorageGRID administration.

For example, this PUT Object request was issued on 22 Jun 2020 and places an object in the `testbucket` bucket.

```
aws s3api --endpoint-url <StorageGRID endpoint> put-object
--bucket testbucket --key obj2test2 --body bktjson.json
```

The success response indicates that the object will expire in 100 days (01 Oct 2020) and that it matched Rule 2 of the lifecycle configuration.

```
{
  *Expiration: "expiry-date=\"Thu, 01 Oct 2020 09:07:49 GMT\", rule-
id=\"rule2\"",
  ETag: "\"9762f8a803bc34f5340579d4446076f7\""
}
```

For example, this HEAD Object request was used to get metadata for the same object in the testbucket bucket.

```
aws s3api --endpoint-url <StorageGRID endpoint> head-object
--bucket testbucket --key obj2test2
```

The success response includes the object's metadata and indicates that the object will expire in 100 days and that it matched Rule 2.

```
{
  "AcceptRanges": "bytes",
  "Expiration": "expiry-date=\"Thu, 01 Oct 2020 09:07:48 GMT\", rule-id=\"rule2\"",
  "LastModified": "2020-06-23T09:07:48+00:00",
  "ContentLength": 921,
  "ETag": "\"9762f8a803bc34f5340579d4446076f7\"",
  "ContentType": "binary/octet-stream",
  "Metadata": {}
}
```

Related information

[Operations on buckets](#)

[Manage objects with ILM](#)

Custom operations on buckets

The StorageGRID system supports custom bucket operations that are added on to the S3 REST API and are specific to the system.

The following table lists the custom bucket operations supported by StorageGRID.

Operation	Description	For more information
GET Bucket consistency	Returns the consistency level being applied to a particular bucket.	GET Bucket consistency request
PUT Bucket consistency	Sets the consistency level applied to a particular bucket.	PUT Bucket consistency request
GET Bucket last access time	Returns whether last access time updates are enabled or disabled for a particular bucket.	GET Bucket last access time request
PUT Bucket last access time	Allows you to enable or disable last access time updates for a particular bucket.	PUT Bucket last access time request

Operation	Description	For more information
DELETE Bucket metadata notification configuration	Deletes the metadata notification configuration XML associated with a particular bucket.	DELETE Bucket metadata notification configuration request
GET Bucket metadata notification configuration	Returns the metadata notification configuration XML associated with a particular bucket.	GET Bucket metadata notification configuration request
PUT Bucket metadata notification configuration	Configures the metadata notification service for a bucket.	PUT Bucket metadata notification configuration request
PUT Bucket modifications for compliance	Deprecated and not supported: You can no longer create new buckets with Compliance enabled.	Deprecated: PUT Bucket request modifications for compliance
GET Bucket compliance	Deprecated but supported: Returns the compliance settings currently in effect for an existing legacy Compliant bucket.	Deprecated: GET Bucket compliance request
PUT Bucket compliance	Deprecated but supported: Allows you to modify the compliance settings for an existing legacy Compliant bucket.	Deprecated: PUT Bucket compliance request

Related information

[S3 operations tracked in the audit logs](#)

Operations on objects

This section describes how the StorageGRID system implements S3 REST API operations for objects.

- [Using S3 Object Lock](#)
- [Using server-side encryption](#)
- [GET Object](#)
- [HEAD Object](#)
- [POST Object restore](#)
- [PUT Object](#)
- [PUT Object - Copy](#)

The following conditions apply to all object operations:

- StorageGRID consistency controls are supported by all operations on objects, with the exception of the following:

- GET Object ACL
- OPTIONS /
- PUT Object legal hold
- PUT Object retention
- Conflicting client requests, such as two clients writing to the same key, are resolved on a “latest-wins” basis. The timing for the “latest-wins” evaluation is based on when the StorageGRID system completes a given request, and not on when S3 clients begin an operation.
- All objects in a StorageGRID bucket are owned by the bucket owner, including objects created by an anonymous user, or by another account.
- Data objects ingested to the StorageGRID system through Swift cannot be accessed through S3.

The following table describes how StorageGRID implements S3 REST API object operations.

Operation	Implementation
DELETE Object	<p data-bbox="815 155 1458 226">Multi-Factor Authentication (MFA) and the response header <code>x-amz-mfa</code> are not supported.</p> <p data-bbox="815 260 1479 562">When processing a DELETE Object request, StorageGRID attempts to immediately remove all copies of the object from all stored locations. If successful, StorageGRID returns a response to the client immediately. If all copies cannot be removed within 30 seconds (for example, because a location is temporarily unavailable), StorageGRID queues the copies for removal and then indicates success to the client.</p> <p data-bbox="815 596 961 632">Versioning</p> <p data-bbox="815 665 1479 873">To remove a specific version, the requestor must be the bucket owner and use the <code>versionId</code> subresource. Using this subresource permanently deletes the version. If the <code>versionId</code> corresponds to a delete marker, the response header <code>x-amz-delete-marker</code> is returned set to <code>true</code>.</p> <ul data-bbox="841 907 1479 1419" style="list-style-type: none"> • If an object is deleted without the <code>versionId</code> subresource on a version enabled bucket, it results in the generation of a delete marker. The <code>versionId</code> for the delete marker is returned using the <code>x-amz-version-id</code> response header, and the <code>x-amz-delete-marker</code> response header is returned set to <code>true</code>. • If an object is deleted without the <code>versionId</code> subresource on a version suspended bucket, it results in a permanent deletion of an already existing 'null' version or a 'null' delete marker, and the generation of a new 'null' delete marker. The <code>x-amz-delete-marker</code> response header is returned set to <code>true</code>. <p data-bbox="815 1453 1474 1524">Note: In certain cases, multiple delete markers might exist for an object.</p>
DELETE Multiple Objects	<p data-bbox="815 1570 1458 1642">Multi-Factor Authentication (MFA) and the response header <code>x-amz-mfa</code> are not supported.</p> <p data-bbox="815 1675 1458 1747">Multiple objects can be deleted in the same request message.</p>

Operation	Implementation
DELETE Object tagging	<p>Uses the <code>tagging</code> subresource to remove all tags from an object. Implemented with all Amazon S3 REST API behavior.</p> <p>Versioning</p> <p>If the <code>versionId</code> query parameter is not specified in the request, the operation deletes all tags from the most recent version of the object in a versioned bucket. If the current version of the object is a delete marker, a “MethodNotAllowed” status is returned with the <code>x-amz-delete-marker</code> response header set to <code>true</code>.</p>
GET Object	GET Object
GET Object ACL	If the necessary access credentials are provided for the account, the operation returns a positive response and the ID, DisplayName, and Permission of the object owner, indicating that the owner has full access to the object.
GET Object legal hold	Using S3 Object Lock
GET Object retention	Using S3 Object Lock
GET Object tagging	<p>Uses the <code>tagging</code> subresource to return all tags for an object. Implemented with all Amazon S3 REST API behavior</p> <p>Versioning</p> <p>If the <code>versionId</code> query parameter is not specified in the request, the operation returns all tags from the most recent version of the object in a versioned bucket. If the current version of the object is a delete marker, a “MethodNotAllowed” status is returned with the <code>x-amz-delete-marker</code> response header set to <code>true</code>.</p>
HEAD Object	HEAD Object
POST Object restore	POST Object restore
PUT Object	PUT Object
PUT Object - Copy	PUT Object - Copy

Operation	Implementation
PUT Object legal hold	Using S3 Object Lock
PUT Object retention	Using S3 Object Lock
PUT Object tagging	<p>Uses the <code>tagging</code> subresource to add a set of tags to an existing object. Implemented with all Amazon S3 REST API behavior</p> <p>Tag updates and ingest behavior</p> <p>When you use PUT Object tagging to update an object's tags, StorageGRID does not re-ingest the object. This means that the option for Ingest Behavior specified in the matching ILM rule is not used. Any changes to object placement that are triggered by the update are made when ILM is re-evaluated by normal background ILM processes.</p> <p>This means that if the ILM rule uses the Strict option for ingest behavior, no action is taken if the required object placements cannot be made (for example, because a newly required location is unavailable). The updated object retains its current placement until the required placement is possible.</p> <p>Resolving conflicts</p> <p>Conflicting client requests, such as two clients writing to the same key, are resolved on a "latest-wins" basis. The timing for the "latest-wins" evaluation is based on when the StorageGRID system completes a given request, and not on when S3 clients begin an operation.</p> <p>Versioning</p> <p>If the <code>versionId</code> query parameter is not specified in the request, the operation add tags to the most recent version of the object in a versioned bucket. If the current version of the object is a delete marker, a "MethodNotAllowed" status is returned with the <code>x-amz-delete-marker</code> response header set to <code>true</code>.</p>

Related information

[Consistency controls](#)

[S3 operations tracked in the audit logs](#)

Using S3 Object Lock

If the global S3 Object Lock setting is enabled for your StorageGRID system, you can create buckets with S3 Object Lock enabled and then specify retain-until-date and legal hold settings for each object version you add to that bucket.

S3 Object Lock allows you to specify object-level settings to prevent objects from being deleted or overwritten for a fixed amount of time or indefinitely.

The StorageGRID S3 Object Lock feature provides a single retention mode that is equivalent to the Amazon S3 compliance mode. By default, a protected object version cannot be overwritten or deleted by any user. The StorageGRID S3 Object Lock feature does not support a governance mode, and it does not allow users with special permissions to bypass retention settings or to delete protected objects.

Enabling S3 Object Lock for a bucket

If the global S3 Object Lock setting is enabled for your StorageGRID system, you can optionally enable S3 Object Lock when you create each bucket. You can use either of these methods:

- Create the bucket using the Tenant Manager.

[Use a tenant account](#)

- Create the bucket using a PUT Bucket request with the `x-amz-bucket-object-lock_enabled` request header.

[Operations on buckets](#)

You cannot add or disable S3 Object Lock after the bucket is created. S3 Object Lock requires bucket versioning, which is enabled automatically when you create the bucket.

A bucket with S3 Object Lock enabled can contain a combination of objects with and without S3 Object Lock settings. StorageGRID does not support default retention for the objects in S3 Object Lock buckets, so the PUT Object Lock Configuration bucket operation is not supported.

Determining if S3 Object Lock is enabled for a bucket

To determine if S3 Object Lock is enabled, use the GET Object Lock Configuration request.

[Operations on buckets](#)

Creating an object with S3 Object Lock settings

To specify S3 Object Lock settings when adding an object version to a bucket that has S3 Object Lock enabled, issue a PUT Object, PUT Object - Copy, or Initiate Multipart Upload request. Use the following request headers.



You must enable S3 Object Lock when you create a bucket. You cannot add or disable S3 Object Lock after a bucket is created.

- `x-amz-object-lock-mode`, which must be COMPLIANCE (case sensitive).



If you specify `x-amz-object-lock-mode`, you must also specify `x-amz-object-lock-retain-until-date`.

- `x-amz-object-lock-retain-until-date`
 - The `retain-until-date` value must be in the format `2020-08-10T21:46:00Z`. Fractional seconds are allowed, but only 3 decimal digits are preserved (milliseconds precision). Other ISO 8601 formats are not allowed.
 - The `retain-until-date` must be in the future.
- `x-amz-object-lock-legal-hold`

If legal hold is ON (case-sensitive), the object is placed under a legal hold. If legal hold is OFF, no legal hold is placed. Any other value results in a 400 Bad Request (InvalidArgument) error.

If you use any of these request headers, be aware of these restrictions:

- The `Content-MD5` request header is required if any `x-amz-object-lock-*` request header is present in the PUT Object request. `Content-MD5` is not required for PUT Object - Copy or Initiate Multipart Upload.
- If the bucket does not have S3 Object Lock enabled and a `x-amz-object-lock-*` request header is present, a 400 Bad Request (InvalidRequest) error is returned.
- The PUT Object request supports the use of `x-amz-storage-class: REDUCED_REDUNDANCY` to match AWS behavior. However, when an object is ingested into a bucket with S3 Object Lock enabled, StorageGRID will always perform a dual-commit ingest.
- A subsequent GET or HEAD Object version response will include the headers `x-amz-object-lock-mode`, `x-amz-object-lock-retain-until-date`, and `x-amz-object-lock-legal-hold`, if configured and if the request sender has the correct `s3:Get*` permissions.
- A subsequent DELETE Object version or DELETE Objects versions request will fail if it is before the `retain-until-date` or if a legal hold is on.

Updating S3 Object Lock settings

If you need to update the legal hold or retention settings for an existing object version, you can perform the following object subresource operations:

- `PUT Object legal-hold`

If the new legal-hold value is ON, the object is placed under a legal hold. If the legal-hold value is OFF, the legal hold is lifted.

- `PUT Object retention`
 - The mode value must be COMPLIANCE (case sensitive).
 - The `retain-until-date` value must be in the format `2020-08-10T21:46:00Z`. Fractional seconds are allowed, but only 3 decimal digits are preserved (milliseconds precision). Other ISO 8601 formats are not allowed.
 - If an object version has an existing `retain-until-date`, you can only increase it. The new value must be in the future.

Related information

[Manage objects with ILM](#)

[Use a tenant account](#)

[PUT Object](#)

[PUT Object - Copy](#)

[Initiate Multipart Upload](#)

[Object versioning](#)

[Amazon Simple Storage Service User Guide: Using S3 Object Lock](#)

Using server-side encryption

Server-side encryption allows you to protect your object data at rest. StorageGRID encrypts the data as it writes the object and decrypts the data when you access the object.

If you want to use server-side encryption, you can choose either of two mutually exclusive options, based on how the encryption keys are managed:

- **SSE (server-side encryption with StorageGRID-managed keys):** When you issue an S3 request to store an object, StorageGRID encrypts the object with a unique key. When you issue an S3 request to retrieve the object, StorageGRID uses the stored key to decrypt the object.
- **SSE-C (server-side encryption with customer-provided keys):** When you issue an S3 request to store an object, you provide your own encryption key. When you retrieve an object, you provide the same encryption key as part of your request. If the two encryption keys match, the object is decrypted and your object data is returned.

While StorageGRID manages all object encryption and decryption operations, you must manage the encryption keys you provide.



The encryption keys you provide are never stored. If you lose an encryption key, you lose the corresponding object.



If an object is encrypted with SSE or SSE-C, any bucket-level or grid-level encryption settings are ignored.

Using SSE

To encrypt an object with a unique key managed by StorageGRID, you use the following request header:

```
x-amz-server-side-encryption
```

The SSE request header is supported by the following object operations:

- PUT Object
- PUT Object - Copy
- Initiate Multipart Upload

Using SSE-C

To encrypt an object with a unique key that you manage, you use three request headers:

Request header	Description
x-amz-server-side-encryption-customer-algorithm	Specify the encryption algorithm. The header value must be AES256.
x-amz-server-side-encryption-customer-key	Specify the encryption key that will be used to encrypt or decrypt the object. The value for the key must be 256-bit, base64-encoded.
x-amz-server-side-encryption-customer-key-MD5	Specify the MD5 digest of the encryption key according to RFC 1321, which is used to ensure the encryption key was transmitted without error. The value for the MD5 digest must be base64-encoded 128-bit.

The SSE-C request headers are supported by the following object operations:

- GET Object
- HEAD Object
- PUT Object
- PUT Object - Copy
- Initiate Multipart Upload
- Upload Part
- Upload Part - Copy

Considerations for using server-side encryption with customer-provided keys (SSE-C)

Before using SSE-C, be aware of the following considerations:

- You must use https.



StorageGRID rejects any requests made over http when using SSE-C. For security considerations, you should consider any key you send accidentally using http to be compromised. Discard the key, and rotate as appropriate.

- The ETag in the response is not the MD5 of the object data.
- You must manage the mapping of encryption keys to objects. StorageGRID does not store encryption keys. You are responsible for tracking the encryption key you provide for each object.
- If your bucket is versioning-enabled, each object version should have its own encryption key. You are responsible for tracking the encryption key used for each object version.
- Because you manage encryption keys on the client side, you must also manage any additional safeguards, such as key rotation, on the client side.



The encryption keys you provide are never stored. If you lose an encryption key, you lose the corresponding object.

- If CloudMirror replication is configured for the bucket, you cannot ingest SSE-C objects. The ingest operation will fail.

Related information

[GET Object](#)

[HEAD Object](#)

[PUT Object](#)

[PUT Object - Copy](#)

[Initiate Multipart Upload](#)

[Upload Part](#)

[Upload Part - Copy](#)

[Amazon S3 Developer Guide: Protecting Data Using Server-Side Encryption with Customer-Provided Encryption Keys \(SSE-C\)](#)

GET Object

You can use the S3 GET Object request to retrieve an object from an S3 bucket.

partNumber request parameter is not supported

The `partNumber` request parameter is not supported for GET Object requests. You cannot perform a GET request to retrieve a specific part of a multipart object. A 501 Not Implemented error is returned with the following message:

```
GET Object by partNumber is not implemented
```

Request headers for server-side encryption with customer-provided encryption keys (SSE-C)

Use all three of the headers if the object is encrypted with a unique key that you provided.

- `x-amz-server-side-encryption-customer-algorithm`: Specify AES256.
- `x-amz-server-side-encryption-customer-key`: Specify your encryption key for the object.
- `x-amz-server-side-encryption-customer-key-MD5`: Specify the MD5 digest of the object's encryption key.



The encryption keys you provide are never stored. If you lose an encryption key, you lose the corresponding object. Before using customer-provided keys to secure object data, review the considerations in “Using server-side encryption.”

UTF-8 characters in user metadata

StorageGRID does not parse or interpret escaped UTF-8 characters in user-defined metadata. GET requests for an object with escaped UTF-8 characters in user-defined metadata do not return the `x-amz-missing-meta` header if the key name or value includes unprintable characters.

Unsupported request header

The following request header is not supported and returns `XNotImplemented`:

- `x-amz-website-redirect-location`

Versioning

If a `versionId` subresource is not specified, the operation fetches the most recent version of the object in a versioned bucket. If the current version of the object is a delete marker, a “Not Found” status is returned with the `x-amz-delete-marker` response header set to `true`.

Behavior of GET Object for Cloud Storage Pool objects

If an object has been stored in a Cloud Storage Pool (see the instructions for managing objects with information lifecycle management), the behavior of a GET Object request depends on the state of the object. See “HEAD Object” for more details.



If an object is stored in a Cloud Storage Pool and one or more copies of the object also exist on the grid, GET Object requests will attempt to retrieve data from the grid, before retrieving it from the Cloud Storage Pool.

State of object	Behavior of GET Object
Object ingested into StorageGRID but not yet evaluated by ILM, or object stored in a traditional storage pool or using erasure coding	200 OK A copy of the object is retrieved.
Object in Cloud Storage Pool but not yet transitioned to a non-retrievable state	200 OK A copy of the object is retrieved.
Object transitioned to a non-retrievable state	403 Forbidden, InvalidObjectState Use a POST Object restore request to restore the object to a retrievable state.
Object in process of being restored from a non-retrievable state	403 Forbidden, InvalidObjectState Wait for the POST Object restore request to complete.
Object fully restored to the Cloud Storage Pool	200 OK A copy of the object is retrieved.

Multipart or segmented objects in a Cloud Storage Pool

If you uploaded a multipart object or if StorageGRID split a large object into segments, StorageGRID determines whether the object is available in the Cloud Storage Pool by sampling a subset of the object's parts or segments. In some cases, a GET Object request might incorrectly return `200 OK` when some parts of the object have already been transitioned to a non-retrievable state or when some parts of the object have not yet been restored.

In these cases:

- The GET Object request might return some data but stop midway through the transfer.
- A subsequent GET Object request might return `403 Forbidden`.

Related information

[Using server-side encryption](#)

[Manage objects with ILM](#)

[POST Object restore](#)

[S3 operations tracked in the audit logs](#)

HEAD Object

You can use the S3 HEAD Object request to retrieve metadata from an object without returning the object itself. If the object is stored in a Cloud Storage Pool, you can use HEAD Object to determine the object's transition state.

Request headers for server-side encryption with customer-provided encryption keys (SSE-C)

Use all three of these headers if the object is encrypted with a unique key that you provided.

- `x-amz-server-side-encryption-customer-algorithm`: Specify AES256.
- `x-amz-server-side-encryption-customer-key`: Specify your encryption key for the object.
- `x-amz-server-side-encryption-customer-key-MD5`: Specify the MD5 digest of the object's encryption key.



The encryption keys you provide are never stored. If you lose an encryption key, you lose the corresponding object. Before using customer-provided keys to secure object data, review the considerations in “Using server-side encryption.”

UTF-8 characters in user metadata

StorageGRID does not parse or interpret escaped UTF-8 characters in user-defined metadata. HEAD requests for an object with escaped UTF-8 characters in user-defined metadata do not return the `x-amz-missing-meta` header if the key name or value includes unprintable characters.

Unsupported request header

The following request header is not supported and returns `XNotImplemented`:

- `x-amz-website-redirect-location`

Response headers for Cloud Storage Pool objects

If the object is stored in a Cloud Storage Pool (see the instructions for managing objects with information lifecycle management), the following response headers are returned:

- `x-amz-storage-class: GLACIER`
- `x-amz-restore`

The response headers provide information about the state of an object as it is moved to a Cloud Storage Pool, optionally transitioned to a non-retrievable state, and restored.

State of object	Response to HEAD object
Object ingested into StorageGRID but not yet evaluated by ILM, or object stored in a traditional storage pool or using erasure coding	200 OK (No special response header is returned.)
Object in Cloud Storage Pool but not yet transitioned to a non-retrievable state	<p>200 OK</p> <p><code>x-amz-storage-class: GLACIER</code></p> <p><code>x-amz-restore: ongoing-request="false", expiry-date="Sat, 23 July 20 2030 00:00:00 GMT"</code></p> <p>Until the object is transitioned to a non-retrievable state, the value for <code>expiry-date</code> is set to some distant time in the future. The exact time of transition is not controlled by the StorageGRID system.</p>
Object has transitioned to non-retrievable state, but at least one copy also exists on the grid	<p>200 OK</p> <p><code>x-amz-storage-class: GLACIER</code></p> <p><code>x-amz-restore: ongoing-request="false", expiry-date="Sat, 23 July 20 2030 00:00:00 GMT"</code></p> <p>The value for <code>expiry-date</code> is set to some distant time in the future.</p> <p>Note: If the copy on the grid is not available (for example, a Storage Node is down), you must issue a POST Object restore request to restore the copy from the Cloud Storage Pool before you can successfully retrieve the object.</p>

State of object	Response to HEAD object
Object transitioned to a non-retrievable state, and no copy exists on the grid	200 OK x-amz-storage-class: GLACIER
Object in process of being restored from a non-retrievable state	200 OK x-amz-storage-class: GLACIER x-amz-restore: ongoing-request="true"
Object fully restored to the Cloud Storage Pool	200 OK x-amz-storage-class: GLACIER x-amz-restore: ongoing-request="false", expiry-date="Sat, 23 July 20 2018 00:00:00 GMT" The expiry-date indicates when the object in the Cloud Storage Pool will be returned to a non-retrievable state.

Multipart or segmented objects in a Cloud Storage Pool

If you uploaded a multipart object or if StorageGRID split a large object into segments, StorageGRID determines whether the object is available in the Cloud Storage Pool by sampling a subset of the object's parts or segments. In some cases, a HEAD Object request might incorrectly return `x-amz-restore: ongoing-request="false"` when some parts of the object have already been transitioned to a non-retrievable state or when some parts of the object have not yet been restored.

Versioning

If a `versionId` subresource is not specified, the operation fetches the most recent version of the object in a versioned bucket. If the current version of the object is a delete marker, a "Not Found" status is returned with the `x-amz-delete-marker` response header set to `true`.

Related information

[Using server-side encryption](#)

[Manage objects with ILM](#)

[POST Object restore](#)

[S3 operations tracked in the audit logs](#)

POST Object restore

You can use the S3 POST Object restore request to restore an object that is stored in a Cloud Storage Pool.

Supported request type

StorageGRID only supports POST Object restore requests to restore an object. It does not support the SELECT type of restoration. Select requests return `XNotImplemented`.

Versioning

Optionally, specify `versionId` to restore a specific version of an object in a versioned bucket. If you do not specify `versionId`, the most recent version of the object is restored

Behavior of POST Object restore on Cloud Storage Pool objects

If an object has been stored in a Cloud Storage Pool (see the instructions for managing objects with information lifecycle management), a POST Object restore request has the following behavior, based on the state of the object. See “HEAD Object” for more details.



If an object is stored in a Cloud Storage Pool and one or more copies of the object also exist on the grid, there is no need to restore the object by issuing a POST Object restore request. Instead, the local copy can be retrieved directly, using a GET Object request.

State of object	Behavior of POST Object restore
Object ingested into StorageGRID but not yet evaluated by ILM, or object is not in a Cloud Storage Pool	403 Forbidden, InvalidObjectState
Object in Cloud Storage Pool but not yet transitioned to a non-retrievable state	200 OK No changes are made. Note: Before an object has been transitioned to a non-retrievable state, you cannot change its expiry-date.
Object transitioned to a non-retrievable state	202 Accepted Restores a retrievable copy of the object to the Cloud Storage Pool for the number of days specified in the request body. At the end of this period, the object is returned to a non-retrievable state. Optionally, use the <code>Tier</code> request element to determine how long the restore job will take to finish (Expedited, Standard, or Bulk). If you do not specify <code>Tier</code> , the <code>Standard</code> tier is used. Attention: If an object has been transitioned to S3 Glacier Deep Archive or the Cloud Storage Pool uses Azure Blob Storage, you cannot restore it using the Expedited tier. The following error is returned 403 Forbidden, InvalidTier: Retrieval option is not supported by this storage class.

State of object	Behavior of POST Object restore
Object in process of being restored from a non-retrievable state	409 Conflict, RestoreAlreadyInProgress
Object fully restored to the Cloud Storage Pool	200 OK Note: If an object has been restored to a retrievable state, you can change its <code>expiry-date</code> by reissuing the POST Object restore request with a new value for <code>Days</code> . The restoration date is updated relative to the time of the request.

Related information

[Manage objects with ILM](#)

[HEAD Object](#)

[S3 operations tracked in the audit logs](#)

PUT Object

You can use the S3 PUT Object request to add an object to a bucket.

Resolving conflicts

Conflicting client requests, such as two clients writing to the same key, are resolved on a “latest-wins” basis. The timing for the “latest-wins” evaluation is based on when the StorageGRID system completes a given request, and not on when S3 clients begin an operation.

Object size

StorageGRID supports objects up to 5 TB in size.

User metadata size

Amazon S3 limits the size of user-defined metadata within each PUT request header to 2 KB. StorageGRID limits user metadata to 24 KiB. The size of user-defined metadata is measured by taking the sum of the number of bytes in the UTF-8 encoding of each key and value.

UTF-8 characters in user metadata

If a request includes (unescaped) UTF-8 values in the key name or value of user-defined metadata, StorageGRID behavior is undefined.

StorageGRID does not parse or interpret escaped UTF-8 characters included in the key name or value of user-defined metadata. Escaped UTF-8 characters are treated as ASCII characters:

- PUT, PUT Object-Copy, GET, and HEAD requests succeed if user-defined metadata includes escaped UTF-8 characters.
- StorageGRID does not return the `x-amz-missing-meta` header if the interpreted value of the key name or value includes unprintable characters.

Object tag limits

You can add tags to new objects when you upload them, or you can add them to existing objects. Both StorageGRID and Amazon S3 support up to 10 tags for each object. Tags associated with an object must have unique tag keys. A tag key can be up to 128 Unicode characters in length and tag values can be up to 256 Unicode characters in length. Key and values are case sensitive.

Object ownership

In StorageGRID, all objects are owned by the bucket owner account, including objects created by a non-owner account or an anonymous user.

Supported request headers

The following request headers are supported:

- Cache-Control
- Content-Disposition
- Content-Encoding

When you specify `aws-chunked` for Content-Encoding StorageGRID does not verify the following items:

- StorageGRID does not verify the `chunk-signature` against the chunk data.
- StorageGRID does not verify the value that you provide for `x-amz-decoded-content-length` against the object.
- Content-Language
- Content-Length
- Content-MD5
- Content-Type
- Expires
- Transfer-Encoding

Chunked transfer encoding is supported if `aws-chunked` payload signing is also used.

- `x-amz-meta-`, followed by a name-value pair containing user-defined metadata.

When specifying the name-value pair for user-defined metadata, use this general format:

```
x-amz-meta-<name>: <value>
```

If you want to use the **User Defined Creation Time** option as the Reference Time for an ILM rule, you must use `creation-time` as the name of the metadata that records when the object was created. For example:

```
x-amz-meta-creation-time: 1443399726
```

The value for `creation-time` is evaluated as seconds since January 1, 1970.



An ILM rule cannot use both a **User Defined Creation Time** for the Reference Time and the Balanced or Strict options for Ingest Behavior. An error is returned when the ILM rule is created.

- `x-amz-tagging`
- S3 Object Lock request headers
 - `x-amz-object-lock-mode`
 - `x-amz-object-lock-retain-until-date`
 - `x-amz-object-lock-legal-hold`

Using S3 Object Lock

- SSE request headers:
 - `x-amz-server-side-encryption`
 - `x-amz-server-side-encryption-customer-key-MD5`
 - `x-amz-server-side-encryption-customer-key`
 - `x-amz-server-side-encryption-customer-algorithm`

S3 REST API supported operations and limitations

Unsupported request headers

The following request headers are not supported:

- The `x-amz-acl` request header is not supported.
- The `x-amz-website-redirect-location` request header is not supported and returns `XNotImplemented`.

Storage class options

The `x-amz-storage-class` request header is supported. The value submitted for `x-amz-storage-class` affects how StorageGRID protects object data during ingest and not how many persistent copies of the object are stored in the StorageGRID system (which is determined by ILM).

If the ILM rule matching an ingested object uses the Strict option for Ingest Behavior, the `x-amz-storage-class` header has no effect.

The following values can be used for `x-amz-storage-class`:

- STANDARD (Default)
 - **Dual commit:** If the ILM rule specifies the Dual commit option for Ingest Behavior, as soon as an object is ingested a second copy of that object is created and distributed to a different Storage Node (dual

commit). When the ILM is evaluated, StorageGRID determines if these initial interim copies satisfy the placement instructions in the rule. If they do not, new object copies might need to be made in different locations and the initial interim copies might need to be deleted.

- **Balanced:** If the ILM rule specifies the Balanced option and StorageGRID cannot immediately make all copies specified in the rule, StorageGRID makes two interim copies on different Storage Nodes.

If StorageGRID can immediately create all object copies specified in the ILM rule (synchronous placement), the `x-amz-storage-class` header has no effect.

- `REDUCED_REDUNDANCY`

- **Dual commit:** If the ILM rule specifies the Dual commit option for Ingest Behavior, StorageGRID creates a single interim copy as the object is ingested (single commit).
- **Balanced:** If the ILM rule specifies the Balanced option, StorageGRID makes a single interim copy only if the system cannot immediately make all copies specified in the rule. If StorageGRID can perform synchronous placement, this header has no effect. The `REDUCED_REDUNDANCY` option is best used when the ILM rule that matches the object creates a single replicated copy. In this case using `REDUCED_REDUNDANCY` eliminates the unnecessary creation and deletion of an extra object copy for every ingest operation.

Using the `REDUCED_REDUNDANCY` option is not recommended in other circumstances.

`REDUCED_REDUNDANCY` increases the risk of object data loss during ingest. For example, you might lose data if the single copy is initially stored on a Storage Node that fails before ILM evaluation can occur.

Attention: Having only one replicated copy for any time period puts data at risk of permanent loss. If only one replicated copy of an object exists, that object is lost if a Storage Node fails or has a significant error. You also temporarily lose access to the object during maintenance procedures such as upgrades.

Specifying `REDUCED_REDUNDANCY` only affects how many copies are created when an object is first ingested. It does not affect how many copies of the object are made when the object is evaluated by the active ILM policy, and does not result in data being stored at lower levels of redundancy in the StorageGRID system.

Note: If you are ingesting an object into a bucket with S3 Object Lock enabled, the `REDUCED_REDUNDANCY` option is ignored. If you are ingesting an object into a legacy Compliant bucket, the `REDUCED_REDUNDANCY` option returns an error. StorageGRID will always perform a dual-commit ingest to ensure that compliance requirements are satisfied.

Request headers for server-side encryption

You can use the following request headers to encrypt an object with server-side encryption. The SSE and SSE-C options are mutually exclusive.

- **SSE:** Use the following header if you want to encrypt the object with a unique key managed by StorageGRID.
 - `x-amz-server-side-encryption`
- **SSE-C:** Use all three of these headers if you want to encrypt the object with a unique key that you provide and manage.
 - `x-amz-server-side-encryption-customer-algorithm`: Specify AES256.
 - `x-amz-server-side-encryption-customer-key`: Specify your encryption key for the new object.

- `x-amz-server-side-encryption-customer-key-MD5`: Specify the MD5 digest of the new object's encryption key.

Attention: The encryption keys you provide are never stored. If you lose an encryption key, you lose the corresponding object. Before using customer-provided keys to secure object data, review the considerations in “Using server-side encryption.”

Note: If an object is encrypted with SSE or SSE-C, any bucket-level or grid-level encryption settings are ignored.

Versioning

If versioning is enabled for a bucket, a unique `versionId` is automatically generated for the version of the object being stored. This `versionId` is also returned in the response using the `x-amz-version-id` response header.

If versioning is suspended, the object version is stored with a null `versionId` and if a null version already exists it will be overwritten.

Related information

[Manage objects with ILM](#)

[Operations on buckets](#)

[S3 operations tracked in the audit logs](#)

[Using server-side encryption](#)

[How client connections can be configured](#)

PUT Object - Copy

You can use the S3 PUT Object - Copy request to create a copy of an object that is already stored in S3. A PUT Object - Copy operation is the same as performing a GET and then a PUT.

Resolving conflicts

Conflicting client requests, such as two clients writing to the same key, are resolved on a “latest-wins” basis. The timing for the “latest-wins” evaluation is based on when the StorageGRID system completes a given request, and not on when S3 clients begin an operation.

Object size

StorageGRID supports objects up to 5 TB in size.

UTF-8 characters in user metadata

If a request includes (unescaped) UTF-8 values in the key name or value of user-defined metadata, StorageGRID behavior is undefined.

StorageGRID does not parse or interpret escaped UTF-8 characters included in the key name or value of user-defined metadata. Escaped UTF-8 characters are treated as ASCII characters:

- Requests succeed if user-defined metadata includes escaped UTF-8 characters.
- StorageGRID does not return the `x-amz-missing-meta` header if the interpreted value of the key name or value includes unprintable characters.

Supported request headers

The following request headers are supported:

- `Content-Type`
- `x-amz-copy-source`
- `x-amz-copy-source-if-match`
- `x-amz-copy-source-if-none-match`
- `x-amz-copy-source-if-unmodified-since`
- `x-amz-copy-source-if-modified-since`
- `x-amz-meta-`, followed by a name-value pair containing user-defined metadata
- `x-amz-metadata-directive`: The default value is `COPY`, which enables you to copy the object and associated metadata.

You can specify `REPLACE` to overwrite the existing metadata when copying the object, or to update the object metadata.

- `x-amz-storage-class`
- `x-amz-tagging-directive`: The default value is `COPY`, which enables you to copy the object and all tags.

You can specify `REPLACE` to overwrite the existing tags when copying the object, or to update the tags.

- S3 Object Lock request headers:
 - `x-amz-object-lock-mode`
 - `x-amz-object-lock-retain-until-date`
 - `x-amz-object-lock-legal-hold`

Using S3 Object Lock

- SSE request headers:
 - `x-amz-copy-source-server-side-encryption-customer-algorithm`
 - `x-amz-copy-source-server-side-encryption-customer-key`
 - `x-amz-copy-source-server-side-encryption-customer-key-MD5`
 - `x-amz-server-side-encryption`
 - `x-amz-server-side-encryption-customer-key-MD5`
 - `x-amz-server-side-encryption-customer-key`
 - `x-amz-server-side-encryption-customer-algorithm`

Unsupported request headers

The following request headers are not supported:

- Cache-Control
- Content-Disposition
- Content-Encoding
- Content-Language
- Expires
- x-amz-website-redirect-location

Storage class options

The `x-amz-storage-class` request header is supported, and affects how many object copies StorageGRID creates if the matching ILM rule specifies an Ingest Behavior of Dual commit or Balanced.

- STANDARD

(Default) Specifies a dual-commit ingest operation when the ILM rule uses the Dual commit option, or when the Balanced option falls back to creating interim copies.

- REDUCED_REDUNDANCY

Specifies a single-commit ingest operation when the ILM rule uses the Dual commit option, or when the Balanced option falls back to creating interim copies.



If you are ingesting an object into a bucket with S3 Object Lock enabled, the REDUCED_REDUNDANCY option is ignored. If you are ingesting an object into a legacy Compliant bucket, the REDUCED_REDUNDANCY option returns an error. StorageGRID will always perform a dual-commit ingest to ensure that compliance requirements are satisfied.

Using x-amz-copy-source in PUT Object - Copy

If the source bucket and key, specified in the `x-amz-copy-source` header, are different from the destination bucket and key, a copy of the source object data is written to the destination.

If the source and destination match, and the `x-amz-metadata-directive` header is specified as REPLACE, the object's metadata is updated with the metadata values supplied in the request. In this case, StorageGRID does not re-ingest the object. This has two important consequences:

- You cannot use PUT Object - Copy to encrypt an existing object in place, or to change the encryption of an existing object in place. If you supply the `x-amz-server-side-encryption` header or the `x-amz-server-side-encryption-customer-algorithm` header, StorageGRID rejects the request and returns XNotImplemented.
- The option for Ingest Behavior specified in the matching ILM rule is not used. Any changes to object placement that are triggered by the update are made when ILM is re-evaluated by normal background ILM processes.

This means that if the ILM rule uses the Strict option for ingest behavior, no action is taken if the required object placements cannot be made (for example, because a newly required location is unavailable). The updated object retains its current placement until the required placement is possible.

Request headers for server-side encryption

If you use server-side encryption, the request headers you provide depend on whether the source object is encrypted and on whether you plan to encrypt the target object.

- If the source object is encrypted using a customer-provided key (SSE-C), you must include the following three headers in the PUT Object - Copy request, so the object can be decrypted and then copied:
 - `x-amz-copy-source-server-side-encryption-customer-algorithm` Specify AES256.
 - `x-amz-copy-source-server-side-encryption-customer-key` Specify the encryption key you provided when you created the source object.
 - `x-amz-copy-source-server-side-encryption-customer-key-MD5`: Specify the MD5 digest you provided when you created the source object.
- If you want to encrypt the target object (the copy) with a unique key that you provide and manage, include the following three headers:
 - `x-amz-server-side-encryption-customer-algorithm`: Specify AES256.
 - `x-amz-server-side-encryption-customer-key`: Specify a new encryption key for the target object.
 - `x-amz-server-side-encryption-customer-key-MD5`: Specify the MD5 digest of the new encryption key.

Attention: The encryption keys you provide are never stored. If you lose an encryption key, you lose the corresponding object. Before using customer-provided keys to secure object data, review the considerations in “Using server-side encryption.”

- If you want to encrypt the target object (the copy) with a unique key managed by StorageGRID (SSE), include this header in the PUT Object - Copy request:
 - `x-amz-server-side-encryption`

Note: The `server-side-encryption` value of the object cannot be updated. Instead, make a copy with a new `server-side-encryption` value using `x-amz-metadata-directive: REPLACE`.

Versioning

If the source bucket is versioned, you can use the `x-amz-copy-source` header to copy the latest version of an object. To copy a specific version of an object, you must explicitly specify the version to copy using the `versionId` subresource. If the destination bucket is versioned, the generated version is returned in the `x-amz-version-id` response header. If versioning is suspended for the target bucket, then `x-amz-version-id` returns a “null” value.

Related information

[Manage objects with ILM](#)

[Using server-side encryption](#)

[S3 operations tracked in the audit logs](#)

Operations for multipart uploads

This section describes how StorageGRID supports operations for multipart uploads.

- [List multipart uploads](#)
- [Initiate Multipart Upload](#)
- [Upload Part](#)
- [Upload Part - Copy](#)
- [Complete Multipart Upload](#)

The following conditions and notes apply to all multipart upload operations:

- You should not exceed 1,000 concurrent multipart uploads to a single bucket because the results of List Multipart Uploads queries for that bucket might return incomplete results.
- StorageGRID enforces AWS size limits for multipart parts. S3 clients must follow these guidelines:
 - Each part in a multipart upload must be between 5 MiB (5,242,880 bytes) and 5 GiB (5,368,709,120 bytes).
 - The last part can be smaller than 5 MiB (5,242,880 bytes).
 - In general, part sizes should be as large as possible. For example, use part sizes of 5 GiB for a 100 GiB object. Since each part is considered a unique object, using large part sizes reduces StorageGRID metadata overhead.
 - For objects smaller than 5 GiB, consider using non-multipart upload instead.
- ILM is evaluated for each part of a multipart object as it is ingested and for the object as a whole when the multipart upload completes, if the ILM rule uses the Strict or Balanced ingest behavior. You should be aware of how this affects object and part placement:
 - If ILM changes while an S3 multipart upload is in progress, when the multipart upload completes some parts of the object might not meet current ILM requirements. Any part that is not placed correctly is queued for ILM re-evaluation, and is moved to the correct location later.
 - When evaluating ILM for a part, StorageGRID filters on the size of the part, not the size of the object. This means that parts of an object can be stored in locations that do not meet ILM requirements for the object as a whole. For example, if a rule specifies that all objects 10 GB or larger are stored at DC1 while all smaller objects are stored at DC2, at ingest each 1 GB part of a 10-part multipart upload is stored at DC2. When ILM is evaluated for the object as a whole, all parts of the object are moved to DC1.
- All of the multipart upload operations support StorageGRID consistency controls.
- As required, you can use server-side encryption with multipart uploads. To use SSE (server-side encryption with StorageGRID-managed keys), you include the `x-amz-server-side-encryption` request header in the Initiate Multipart Upload request only. To use SSE-C (server-side encryption with customer-provided keys), you specify the same three encryption key request headers in the Initiate Multipart Upload request and in each subsequent Upload Part request.

Operation	Implementation
List Multipart Uploads	See List Multipart Uploads

Operation	Implementation
Initiate Multipart Upload	See Initiate Multipart Upload
Upload Part	See Upload Part
Upload Part - Copy	See Upload Part - Copy
Complete Multipart Upload	See Complete Multipart Upload
Abort Multipart Upload	Implemented with all Amazon S3 REST API behavior
List Parts	Implemented with all Amazon S3 REST API behavior

Related information

[Consistency controls](#)

[Using server-side encryption](#)

List Multipart Uploads

The List Multipart Uploads operation lists in-progress multipart uploads for a bucket.

The following request parameters are supported:

- `encoding-type`
- `max-uploads`
- `key-marker`
- `prefix`
- `upload-id-marker`

The `delimiter` request parameter is not supported.

Versioning

Multipart upload consists of separate operations for initiating the upload, listing uploads, uploading parts, assembling the uploaded parts, and completing the upload. When the Complete Multipart Upload operation is performed, that is the point when objects are created (and versioned if applicable).

Initiate Multipart Upload

The Initiate Multipart Upload operation initiates a multipart upload for an object, and returns an upload ID.

The `x-amz-storage-class` request header is supported. The value submitted for `x-amz-storage-class` affects how StorageGRID protects object data during ingest and not how many persistent copies of the object are stored in the StorageGRID system (which is determined by ILM).

If the ILM rule matching an ingested object uses the Strict option for Ingest Behavior, the `x-amz-storage-class` header has no effect.

The following values can be used for `x-amz-storage-class`:

- **STANDARD (Default)**
 - **Dual commit:** If the ILM rule specifies the Dual commit option for Ingest Behavior, as soon as an object is ingested a second copy of that object is created and distributed to a different Storage Node (dual commit). When the ILM is evaluated, StorageGRID determines if these initial interim copies satisfy the placement instructions in the rule. If they do not, new object copies might need to be made in different locations and the initial interim copies might need to be deleted.
 - **Balanced:** If the ILM rule specifies the Balanced option and StorageGRID cannot immediately make all copies specified in the rule, StorageGRID makes two interim copies on different Storage Nodes.

If StorageGRID can immediately create all object copies specified in the ILM rule (synchronous placement), the `x-amz-storage-class` header has no effect.

- **REDUCED_REDUNDANCY**
 - **Dual commit:** If the ILM rule specifies the Dual commit option for Ingest Behavior, StorageGRID creates a single interim copy as the object is ingested (single commit).
 - **Balanced:** If the ILM rule specifies the Balanced option, StorageGRID makes a single interim copy only if the system cannot immediately make all copies specified in the rule. If StorageGRID can perform synchronous placement, this header has no effect. The `REDUCED_REDUNDANCY` option is best used when the ILM rule that matches the object creates a single replicated copy. In this case using `REDUCED_REDUNDANCY` eliminates the unnecessary creation and deletion of an extra object copy for every ingest operation.

Using the `REDUCED_REDUNDANCY` option is not recommended in other circumstances.

`REDUCED_REDUNDANCY` increases the risk of object data loss during ingest. For example, you might lose data if the single copy is initially stored on a Storage Node that fails before ILM evaluation can occur.

Attention: Having only one replicated copy for any time period puts data at risk of permanent loss. If only one replicated copy of an object exists, that object is lost if a Storage Node fails or has a significant error. You also temporarily lose access to the object during maintenance procedures such as upgrades.

Specifying `REDUCED_REDUNDANCY` only affects how many copies are created when an object is first ingested. It does not affect how many copies of the object are made when the object is evaluated by the active ILM policy, and does not result in data being stored at lower levels of redundancy in the StorageGRID system.

Note: If you are ingesting an object into a bucket with S3 Object Lock enabled, the `REDUCED_REDUNDANCY` option is ignored. If you are ingesting an object into a legacy Compliant bucket, the `REDUCED_REDUNDANCY` option returns an error. StorageGRID will always perform a dual-commit ingest to ensure that compliance requirements are satisfied.

The following request headers are supported:

- `Content-Type`
- `x-amz-meta-`, followed by a name-value pair containing user-defined metadata

When specifying the name-value pair for user-defined metadata, use this general format:

```
x-amz-meta-_name_: `value`
```

If you want to use the **User Defined Creation Time** option as the Reference Time for an ILM rule, you must use `creation-time` as the name of the metadata that records when the object was created. For example:

```
x-amz-meta-creation-time: 1443399726
```

The value for `creation-time` is evaluated as seconds since January 1, 1970.



Adding `creation-time` as user-defined metadata is not allowed if you are adding an object to a bucket that has legacy Compliance enabled. An error will be returned.

- S3 Object Lock request headers:
 - `x-amz-object-lock-mode`
 - `x-amz-object-lock-retain-until-date`
 - `x-amz-object-lock-legal-hold`

Using S3 Object Lock

- SSE request headers:
 - `x-amz-server-side-encryption`
 - `x-amz-server-side-encryption-customer-key-MD5`
 - `x-amz-server-side-encryption-customer-key`
 - `x-amz-server-side-encryption-customer-algorithm`

S3 REST API supported operations and limitations



For information on how StorageGRID handles UTF-8 characters, see the documentation for PUT Object.

Request headers for server-side encryption

You can use the following request headers to encrypt a multipart object with server-side encryption. The SSE and SSE-C options are mutually exclusive.

- **SSE:** Use the following header in the Initiate Multipart Upload request if you want to encrypt the object with a unique key managed by StorageGRID. Do not specify this header in any of the Upload Part requests.
 - `x-amz-server-side-encryption`
- **SSE-C:** Use all three of these headers in the Initiate Multipart Upload request (and in each subsequent Upload Part request) if you want to encrypt the object with a unique key that you provide and manage.
 - `x-amz-server-side-encryption-customer-algorithm`: Specify AES256.

- `x-amz-server-side-encryption-customer-key`: Specify your encryption key for the new object.
- `x-amz-server-side-encryption-customer-key-MD5`: Specify the MD5 digest of the new object's encryption key.

Attention: The encryption keys you provide are never stored. If you lose an encryption key, you lose the corresponding object. Before using customer-provided keys to secure object data, review the considerations in “Using server-side encryption.”

Unsupported request headers

The following request header is not supported and returns `XNotImplemented`

- `x-amz-website-redirect-location`

Versioning

Multipart upload consists of separate operations for initiating the upload, listing uploads, uploading parts, assembling the uploaded parts, and completing the upload. Objects are created (and versioned if applicable) when the Complete Multipart Upload operation is performed.

Related information

[Manage objects with ILM](#)

[Using server-side encryption](#)

[PUT Object](#)

Upload Part

The Upload Part operation uploads a part in a multipart upload for an object.

Supported request headers

The following request headers are supported:

- `Content-Length`
- `Content-MD5`

Request headers for server-side encryption

If you specified SSE-C encryption for the Initiate Multipart Upload request, you must also include the following request headers in each Upload Part request:

- `x-amz-server-side-encryption-customer-algorithm`: Specify `AES256`.
- `x-amz-server-side-encryption-customer-key`: Specify the same encryption key that you provided in the Initiate Multipart Upload request.
- `x-amz-server-side-encryption-customer-key-MD5`: Specify the same MD5 digest that you provided in the Initiate Multipart Upload request.



The encryption keys you provide are never stored. If you lose an encryption key, you lose the corresponding object. Before using customer-provided keys to secure object data, review the considerations in “Using server-side encryption.”

Versioning

Multipart upload consists of separate operations for initiating the upload, listing uploads, uploading parts, assembling the uploaded parts, and completing the upload. Objects are created (and versioned if applicable) when the Complete Multipart Upload operation is performed.

Related information

[Using server-side encryption](#)

Upload Part - Copy

The Upload Part - Copy operation uploads a part of an object by copying data from an existing object as the data source.

The Upload Part - Copy operation is implemented with all Amazon S3 REST API behavior.

This request reads and writes the object data specified in `x-amz-copy-source-range` within the StorageGRID system.

The following request headers are supported:

- `x-amz-copy-source-if-match`
- `x-amz-copy-source-if-none-match`
- `x-amz-copy-source-if-unmodified-since`
- `x-amz-copy-source-if-modified-since`

Request headers for server-side encryption

If you specified SSE-C encryption for the Initiate Multipart Upload request, you must also include the following request headers in each Upload Part - Copy request:

- `x-amz-server-side-encryption-customer-algorithm`: Specify AES256.
- `x-amz-server-side-encryption-customer-key`: Specify the same encryption key that you provided in the Initiate Multipart Upload request.
- `x-amz-server-side-encryption-customer-key-MD5`: Specify the same MD5 digest that you provided in the Initiate Multipart Upload request.

If the source object is encrypted using a customer-provided key (SSE-C), you must include the following three headers in the Upload Part - Copy request, so the object can be decrypted and then copied:

- `x-amz-copy-source-server-side-encryption-customer-algorithm`: Specify AES256.
- `x-amz-copy-source-server-side-encryption-customer-key`: Specify the encryption key you provided when you created the source object.
- `x-amz-copy-source-server-side-encryption-customer-key-MD5`: Specify the MD5 digest you provided when you created the source object.



The encryption keys you provide are never stored. If you lose an encryption key, you lose the corresponding object. Before using customer-provided keys to secure object data, review the considerations in “Using server-side encryption.”

Versioning

Multipart upload consists of separate operations for initiating the upload, listing uploads, uploading parts, assembling the uploaded parts, and completing the upload. Objects are created (and versioned if applicable) when the Complete Multipart Upload operation is performed.

Complete Multipart Upload

The Complete Multipart Upload operation completes a multipart upload of an object by assembling the previously uploaded parts.

Resolving conflicts

Conflicting client requests, such as two clients writing to the same key, are resolved on a “latest-wins” basis. The timing for the “latest-wins” evaluation is based on when the StorageGRID system completes a given request, and not on when S3 clients begin an operation.

Object size

StorageGRID supports objects up to 5 TB in size.

Request headers

The `x-amz-storage-class` request header is supported, and affects how many object copies StorageGRID creates if the matching ILM rule specifies an Ingest Behavior of Dual commit or Balanced.

- STANDARD

(Default) Specifies a dual-commit ingest operation when the ILM rule uses the Dual commit option, or when the Balanced option falls back to creating interim copies.

- REDUCED_REDUNDANCY

Specifies a single-commit ingest operation when the ILM rule uses the Dual commit option, or when the Balanced option falls back to creating interim copies.



If you are ingesting an object into a bucket with S3 Object Lock enabled, the REDUCED_REDUNDANCY option is ignored. If you are ingesting an object into a legacy Compliant bucket, the REDUCED_REDUNDANCY option returns an error. StorageGRID will always perform a dual-commit ingest to ensure that compliance requirements are satisfied.



If a multipart upload is not completed within 15 days, the operation is marked as inactive and all associated data is deleted from the system.



The `ETag` value returned is not an MD5 sum of the data, but follows the Amazon S3 API implementation of the `ETag` value for multipart objects.

Versioning

This operation completes a multipart upload. If versioning is enabled for a bucket, the object version is created upon completion of the multipart upload.

If versioning is enabled for a bucket, a unique `versionId` is automatically generated for the version of the object being stored. This `versionId` is also returned in the response using the `x-amz-version-id` response header.

If versioning is suspended, the object version is stored with a null `versionId` and if a null version already exists it will be overwritten.



When versioning is enabled for a bucket, completing a multipart upload always creates a new version, even if there are concurrent multipart uploads completed on the same object key. When versioning is not enabled for a bucket, it is possible to initiate a multipart upload and then have another multipart upload initiate and complete first on the same object key. On non-versioned buckets, the multipart upload that completes last takes precedence.

Failed replication, notification, or metadata notification

If the bucket where the multipart upload occurs is configured for a platform service, multipart upload succeeds even if the associated replication or notification action fails.

If this occurs, an alarm is raised in the Grid Manager on Total Events (SMTT). The Last Event message displays “Failed to publish notifications for bucket-nameobject key” for the last object whose notification failed. (To see this message, select **Nodes** > **Storage Node** > **Events**. View Last Event at the top of the table.) Event messages are also listed in `/var/local/log/bycast-err.log`.

A tenant can trigger the failed replication or notification by updating the object’s metadata or tags. A tenant can resubmit the existing values to avoid making unwanted changes.

Related information

[Manage objects with ILM](#)

Error responses

The StorageGRID system supports all standard S3 REST API error responses that apply. In addition, the StorageGRID implementation adds several custom responses.

Supported S3 API error codes

Name	HTTP status
AccessDenied	403 Forbidden
BadDigest	400 Bad Request
BucketAlreadyExists	409 Conflict
BucketNotEmpty	409 Conflict

Name	HTTP status
IncompleteBody	400 Bad Request
InternalServerError	500 Internal Server Error
InvalidAccessKeyId	403 Forbidden
InvalidArgument	400 Bad Request
InvalidBucketName	400 Bad Request
InvalidBucketState	409 Conflict
InvalidDigest	400 Bad Request
InvalidEncryptionAlgorithmError	400 Bad Request
InvalidPart	400 Bad Request
InvalidPartOrder	400 Bad Request
InvalidRange	416 Requested Range Not Satisfiable
InvalidRequest	400 Bad Request
InvalidStorageClass	400 Bad Request
InvalidTag	400 Bad Request
InvalidURI	400 Bad Request
KeyTooLong	400 Bad Request
MalformedXML	400 Bad Request
MetadataTooLarge	400 Bad Request
MethodNotAllowed	405 Method Not Allowed
MissingContentLength	411 Length Required
MissingRequestBodyError	400 Bad Request
MissingSecurityHeader	400 Bad Request

Name	HTTP status
NoSuchBucket	404 Not Found
NoSuchKey	404 Not Found
NoSuchUpload	404 Not Found
NotImplemented	501 Not Implemented
NoSuchBucketPolicy	404 Not Found
ObjectLockConfigurationNotFound	404 Not Found
PreconditionFailed	412 Precondition Failed
RequestTimeTooSkewed	403 Forbidden
ServiceUnavailable	503 Service Unavailable
SignatureDoesNotMatch	403 Forbidden
TooManyBuckets	400 Bad Request
UserKeyMustBeSpecified	400 Bad Request

StorageGRID custom error codes

Name	Description	HTTP status
XBucketLifecycleNotAllowed	Bucket lifecycle configuration is not allowed in a legacy Compliant bucket	400 Bad Request
XBucketPolicyParseException	Failed to parse received bucket policy JSON.	400 Bad Request
XComplianceConflict	Operation denied because of legacy Compliance settings.	403 Forbidden
XComplianceReducedRedundancyForbidden	Reduced redundancy is not allowed in legacy Compliant bucket	400 Bad Request
XMaxBucketPolicyLengthExceeded	Your policy exceeds the maximum allowed bucket policy length.	400 Bad Request

Name	Description	HTTP status
XMissingInternalRequestHeader	Missing a header of an internal request.	400 Bad Request
XNoSuchBucketCompliance	The specified bucket does not have legacy Compliance enabled.	404 Not Found
XNotAcceptable	The request contains one or more accept headers that could not be satisfied.	406 Not Acceptable
XNotImplemented	The request you provided implies functionality that is not implemented.	501 Not Implemented

StorageGRID S3 REST API operations

There are operations added on to the S3 REST API that are specific to StorageGRID system.

GET Bucket consistency request

The GET Bucket consistency request allows you to determine the consistency level being applied to a particular bucket.

The default consistency controls are set to guarantee read-after-write for newly created objects.

You must have the `s3:GetBucketConsistency` permission, or be account root, to complete this operation.

Request example

```
GET /bucket?x-ntap-sg-consistency HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization string</em>
Host: <em>host</em>
```

Response

In the response XML, `<Consistency>` will return one of the following values:

Consistency control	Description
all	All nodes receive the data immediately, or the request will fail.
strong-global	Guarantees read-after-write consistency for all client requests across all sites.

Consistency control	Description
strong-site	Guarantees read-after-write consistency for all client requests within a site.
read-after-new-write	<p>(Default) Provides read-after-write consistency for new objects and eventual consistency for object updates. Offers high availability and data protection guarantees. Matches Amazon S3 consistency guarantees.</p> <p>Note: If your application uses HEAD requests on objects that do not exist, you might receive a high number of 500 Internal Server errors if one or more Storage Nodes are unavailable. To prevent these errors, set the consistency control to “available” unless you require consistency guarantees similar to Amazon S3.</p>
available (eventual consistency for HEAD operations)	Behaves the same as the “read-after-new-write” consistency level, but only provides eventual consistency for HEAD operations. Offers higher availability for HEAD operations than “read-after-new-write” if Storage Nodes are unavailable. Differs from Amazon S3 consistency guarantees for HEAD operations only.

Response example

```

HTTP/1.1 200 OK
Date: Fri, 18 Sep 2020 01:02:18 GMT
Connection: CLOSE
Server: StorageGRID/11.5.0
x-amz-request-id: 12345
Content-Length: 127
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<Consistency xmlns="http://s3.storagegrid.com/doc/2015-02-01/">read-after-
new-write</Consistency>

```

Related information

[Consistency controls](#)

PUT Bucket consistency request

The PUT Bucket consistency request allows you to specify the consistency level to apply to operations performed on a bucket.

The default consistency controls are set to guarantee read-after-write for newly created objects.

You must have the `s3:PutBucketConsistency` permission, or be account root, to complete this operation.

Request

The `x-ntap-sg-consistency` parameter must contain one of the following values:

Consistency control	Description
all	All nodes receive the data immediately, or the request will fail.
strong-global	Guarantees read-after-write consistency for all client requests across all sites.
strong-site	Guarantees read-after-write consistency for all client requests within a site.
read-after-new-write	<p>(Default) Provides read-after-write consistency for new objects and eventual consistency for object updates. Offers high availability and data protection guarantees. Matches Amazon S3 consistency guarantees.</p> <p>Note: If your application uses HEAD requests on objects that do not exist, you might receive a high number of 500 Internal Server errors if one or more Storage Nodes are unavailable. To prevent these errors, set the consistency control to “available” unless you require consistency guarantees similar to Amazon S3.</p>
available (eventual consistency for HEAD operations)	Behaves the same as the “read-after-new-write” consistency level, but only provides eventual consistency for HEAD operations. Offers higher availability for HEAD operations than “read-after-new-write” if Storage Nodes are unavailable. Differs from Amazon S3 consistency guarantees for HEAD operations only.

Note: In general, you should use the “read-after-new-write” consistency control value. If requests are not working correctly, change the application client behavior if possible. Or, configure the client to specify the consistency control for each API request. Set the consistency control at the bucket level only as a last resort.

Request example

```
PUT /bucket?x-ntap-sg-consistency=strong-global HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization string</em>
Host: <em>host</em>
```

Related information

[Consistency controls](#)

GET Bucket last access time request

The GET Bucket last access time request allows you to determine if last access time updates are enabled or disabled for individual buckets.

You must have the `s3:GetBucketLastAccessTime` permission, or be account root, to complete this operation.

Request example

```
GET /bucket?x-ntap-sg-lastaccesstime HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization string</em>
Host: <em>host</em>
```

Response example

This example shows that last access time updates are enabled for the bucket.

```
HTTP/1.1 200 OK
Date: Sat, 29 Nov 2015 01:02:18 GMT
Connection: CLOSE
Server: StorageGRID/10.3.0
x-amz-request-id: 12345
Content-Length: 127
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<LastAccessTime xmlns="http://s3.storagegrid.com/doc/2015-02-01/">enabled
</LastAccessTime>
```

PUT Bucket last access time request

The PUT Bucket last access time request allows you to enable or disable last access time updates for individual buckets. Disabling last access time updates improves performance, and is the default setting for all buckets created with version 10.3.0, or later.

You must have the `s3:PutBucketLastAccessTime` permission for a bucket, or be account root, to complete this operation.



Starting with StorageGRID version 10.3, updates to last access time are disabled by default for all new buckets. If you have buckets that were created using an earlier version of StorageGRID and you want to match the new default behavior, you must explicitly disable last access time updates for each of those earlier buckets. You can enable or disable updates to last access time using the PUT Bucket last access time request, the **S3 > Buckets > Change Last Access Setting** check box in the Tenant Manager, or the Tenant Management API.

If last access time updates are disabled for a bucket, the following behavior is applied to operations on the bucket:

- GET Object, GET Object ACL, GET Object Tagging, and HEAD Object requests do not update last access time. The object is not added to queues for information lifecycle management (ILM) evaluation.
- PUT Object - Copy and PUT Object Tagging requests that update only the metadata also update last access time. The object is added to queues for ILM evaluation.
- If updates to last access time are disabled for the source bucket, PUT Object - Copy requests do not update last access time for the source bucket. The object that was copied is not added to queues for ILM evaluation for the source bucket. However, for the destination, PUT Object - Copy requests always update last access time. The copy of the object is added to queues for ILM evaluation.
- Complete Multipart Upload requests update last access time. The completed object is added to queues for ILM evaluation.

Request examples

This example enables last access time for a bucket.

```
PUT /bucket?x-ntap-sg-lastaccesstime=enabled HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization string</em>
Host: <em>host</em>
```

This example disables last access time for a bucket.

```
PUT /bucket?x-ntap-sg-lastaccesstime=disabled HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization string</em>
Host: <em>host</em>
```

Related information

[Use a tenant account](#)

DELETE Bucket metadata notification configuration request

The DELETE Bucket metadata notification configuration request allows you to disable the search integration service for individual buckets by deleting the configuration XML.

You must have the `s3:DeleteBucketMetadataNotification` permission for a bucket, or be account root, to complete this operation.

Request example

This example shows disabling the search integration service for a bucket.

```
DELETE /test1?x-ntap-sg-metadata-notification HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization string</em>
Host: <em>host</em>
```

GET Bucket metadata notification configuration request

The GET Bucket metadata notification configuration request allows you to retrieve the configuration XML used to configure search integration for individual buckets.

You must have the `s3:GetBucketMetadataNotification` permission, or be account root, to complete this operation.

Request example

This request retrieves the metadata notification configuration for the bucket named `bucket`.

```
GET /bucket?x-ntap-sg-metadata-notification HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization string</em>
Host: <em>host</em>
```

Response

The response body includes the metadata notification configuration for the bucket. The metadata notification configuration lets you determine how the bucket is configured for search integration. That is, it allows you to determine which objects are indexed, and which endpoints their object metadata is being sent to.

```

<MetadataNotificationConfiguration>
  <Rule>
    <ID>Rule-1</ID>
    <Status>rule-status</Status>
    <Prefix>key-prefix</Prefix>
    <Destination>
      <Urn>arn:aws:es:_region:account-
ID_:domain/_mydomain/myindex/mytype_</Urn>
    </Destination>
  </Rule>
  <Rule>
    <ID>Rule-2</ID>
    ...
  </Rule>
  ...
</MetadataNotificationConfiguration>

```

Each metadata notification configuration includes one or more rules. Each rule specifies the objects that it applies to and the destination where StorageGRID should send object metadata. Destinations must be specified using the URN of a StorageGRID endpoint.

Name	Description	Required
MetadataNotificationConfiguration	<p>Container tag for rules used to specify the objects and destination for metadata notifications.</p> <p>Contains one or more Rule elements.</p>	Yes
Rule	<p>Container tag for a rule that identifies the objects whose metadata should be added to a specified index.</p> <p>Rules with overlapping prefixes are rejected.</p> <p>Included in the MetadataNotificationConfiguration element.</p>	Yes
ID	<p>Unique identifier for the rule.</p> <p>Included in the Rule element.</p>	No

Name	Description	Required
Status	<p>Status can be 'Enabled' or 'Disabled'. No action is taken for rules that are disabled.</p> <p>Included in the Rule element.</p>	Yes
Prefix	<p>Objects that match the prefix are affected by the rule, and their metadata is sent to the specified destination.</p> <p>To match all objects, specify an empty prefix.</p> <p>Included in the Rule element.</p>	Yes
Destination	<p>Container tag for the destination of a rule.</p> <p>Included in the Rule element.</p>	Yes

Name	Description	Required
Urn	<p>URN of the destination where object metadata is sent. Must be the URN of a StorageGRID endpoint with the following properties:</p> <ul style="list-style-type: none"> • <code>es</code> must be the third element. • The URN must end with the index and type where the metadata is stored, in the form <code>domain-name/myindex/mytype</code>. <p>Endpoints are configured using the Tenant Manager or Tenant Management API. They take the following form:</p> <ul style="list-style-type: none"> • <code>arn:aws:es:_region:account-ID_:domain/mydomain/myindex/mytype</code> • <code>urn:mysite:es:::mydomain/myindex/mytype</code> <p>The endpoint must be configured before the configuration XML is submitted, or configuration will fail with a 404 error.</p> <p>Urn is included in the Destination element.</p>	Yes

Response example

The XML included between the `<MetadataNotificationConfiguration>` and `</MetadataNotificationConfiguration>` tags shows how integration with a search integration endpoint is configured for the bucket. In this example, object metadata is being sent to an Elasticsearch index named `current` and type named `2017` that is hosted in an AWS domain named `records`.

```
HTTP/1.1 200 OK
Date: Thu, 20 Jul 2017 18:24:05 GMT
Connection: KEEP-ALIVE
Server: StorageGRID/11.0.0
x-amz-request-id: 3832973499
Content-Length: 264
Content-Type: application/xml

<MetadataNotificationConfiguration>
  <Rule>
    <ID>Rule-1</ID>
    <Status>Enabled</Status>
    <Prefix>2017</Prefix>
    <Destination>
      <Urn>arn:aws:es:us-east-
1:3333333:domain/records/current/2017</Urn>
    </Destination>
  </Rule>
</MetadataNotificationConfiguration>
```

Related information

[Use a tenant account](#)

PUT Bucket metadata notification configuration request

The PUT Bucket metadata notification configuration request allows you to enable the search integration service for individual buckets. The metadata notification configuration XML that you supply in the request body specifies the objects whose metadata is sent to the destination search index.

You must have the `s3:PutBucketMetadataNotification` permission for a bucket, or be account root, to complete this operation.

Request

The request must include the metadata notification configuration in the request body. Each metadata notification configuration includes one or more rules. Each rule specifies the objects that it applies to, and the destination where StorageGRID should send object metadata.

Objects can be filtered on the prefix of the object name. For example, you could send metadata for objects with the prefix `/images` to one destination, and objects with the prefix `/videos` to another.

Configurations that have overlapping prefixes are not valid, and are rejected when they are submitted. For example, a configuration that included one rule for for objects with the prefix `test` and a second rule for objects with the prefix `test2` would not be allowed.

Destinations must be specified using the URN of a StorageGRID endpoint. The endpoint must exist when the metadata notification configuration is submitted, or the request fails as a 400 Bad Request. The error message states: `Unable to save the metadata notification (search) policy. The specified endpoint URN does not exist: URN.`

```

<MetadataNotificationConfiguration>
  <Rule>
    <ID>Rule-1</ID>
    <Status>rule-status</Status>
    <Prefix>key-prefix</Prefix>
    <Destination>
      <Urn>arn:aws:es:region:account-
ID:domain/mydomain/myindex/mytype</Urn>
    </Destination>
  </Rule>
  <Rule>
    <ID>Rule-2</ID>
    ...
  </Rule>
  ...
</MetadataNotificationConfiguration>

```

The table describes the elements in the metadata notification configuration XML.

Name	Description	Required
MetadataNotificationConfiguration	<p>Container tag for rules used to specify the objects and destination for metadata notifications.</p> <p>Contains one or more Rule elements.</p>	Yes
Rule	<p>Container tag for a rule that identifies the objects whose metadata should be added to a specified index.</p> <p>Rules with overlapping prefixes are rejected.</p> <p>Included in the MetadataNotificationConfiguration element.</p>	Yes
ID	<p>Unique identifier for the rule.</p> <p>Included in the Rule element.</p>	No

Name	Description	Required
Status	<p>Status can be 'Enabled' or 'Disabled'. No action is taken for rules that are disabled.</p> <p>Included in the Rule element.</p>	Yes
Prefix	<p>Objects that match the prefix are affected by the rule, and their metadata is sent to the specified destination.</p> <p>To match all objects, specify an empty prefix.</p> <p>Included in the Rule element.</p>	Yes
Destination	<p>Container tag for the destination of a rule.</p> <p>Included in the Rule element.</p>	Yes

Name	Description	Required
Urn	<p>URN of the destination where object metadata is sent. Must be the URN of a StorageGRID endpoint with the following properties:</p> <ul style="list-style-type: none"> • es must be the third element. • The URN must end with the index and type where the metadata is stored, in the form domain-name/myindex/mytype. <p>Endpoints are configured using the Tenant Manager or Tenant Management API. They take the following form:</p> <ul style="list-style-type: none"> • arn:aws:es:region:account-ID:domain/mydomain/myindex/mytype • urn:mystore:es:::mydomain/myindex/mytype <p>The endpoint must be configured before the configuration XML is submitted, or configuration will fail with a 404 error.</p> <p>Urn is included in the Destination element.</p>	Yes

Request examples

This example shows enabling search integration for a bucket. In this example, object metadata for all objects is sent to the same destination.

```

PUT /test1?x-ntap-sg-metadata-notification HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization string</em>
Host: <em>host</em>

<MetadataNotificationConfiguration>
  <Rule>
    <ID>Rule-1</ID>
    <Status>Enabled</Status>
    <Prefix></Prefix>
    <Destination>
      <Urn>urn:sgws:es::sgws-notifications/test1/all</Urn>
    </Destination>
  </Rule>
</MetadataNotificationConfiguration>

```

In this example, object metadata for objects that match the prefix `/images` is sent to one destination, while object metadata for objects that match the prefix `/videos` is sent to a second destination.

```

PUT /graphics?x-ntap-sg-metadata-notification HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization string</em>
Host: <em>host</em>

<MetadataNotificationConfiguration>
  <Rule>
    <ID>Images-rule</ID>
    <Status>Enabled</Status>
    <Prefix>/images</Prefix>
    <Destination>
      <Urn>arn:aws:es:us-east-1:3333333:domain/es-
domain/graphics/imagetype</Urn>
    </Destination>
  </Rule>
  <Rule>
    <ID>Videos-rule</ID>
    <Status>Enabled</Status>
    <Prefix>/videos</Prefix>
    <Destination>
      <Urn>arn:aws:es:us-west-1:22222222:domain/es-
domain/graphics/videotype</Urn>
    </Destination>
  </Rule>
</MetadataNotificationConfiguration>

```

Related information
[Use a tenant account](#)

JSON generated by the search integration service

When you enable the search integration service for a bucket, a JSON document is generated and sent to the destination endpoint each time object metadata or tags are added, updated, or deleted.

This example shows an example of the JSON that could be generated when an object with the key SGWS/Tagging.txt is created in a bucket named test. The test bucket is not versioned, so the versionId tag is empty.

```
{
  "bucket": "test",
  "key": "SGWS/Tagging.txt",
  "versionId": "",
  "accountId": "86928401983529626822",
  "size": 38,
  "md5": "3d6c7634a85436eee06d43415012855",
  "region": "us-east-1"
  "metadata": {
    "age": "25"
  },
  "tags": {
    "color": "yellow"
  }
}
```

Object metadata included in metadata notifications

The table lists all the fields that are included in the JSON document that is sent to the destination endpoint when search integration is enabled.

The document name includes the bucket name, object name, and version ID if present.

Type	Item name	Description
Bucket and object information	bucket	Name of the bucket
Bucket and object information	key	Object key name
Bucket and object information	versionID	Object version, for objects in versioned buckets
Bucket and object information	region	Bucket region, for example us-east-1

Type	Item name	Description
System metadata	size	Object size (in bytes) as visible to an HTTP client
System metadata	md5	Object hash
User metadata	metadata <i>key:value</i>	All user metadata for the object, as key-value pairs
Tags	tags <i>key:value</i>	All object tags defined for the object, as key-value pairs

Note: For tags and user metadata, StorageGRID passes dates and numbers to Elasticsearch as strings or as S3 event notifications. To configure Elasticsearch to interpret these strings as dates or numbers, follow the Elasticsearch instructions for dynamic field mapping and for mapping date formats. You must enable the dynamic field mappings on the index before you configure the search integration service. After a document is indexed, you cannot edit the document's field types in the index.

GET Storage Usage request

The GET Storage Usage request tells you the total amount of storage in use by an account, and for each bucket associated with the account.

The amount of storage used by an account and its buckets can be obtained by a modified GET Service request with the `x-ntap-sg-usage` query parameter. Bucket storage usage is tracked separately from the PUT and DELETE requests processed by the system. There might be some delay before the usage values match the expected values based on the processing of requests, particularly if the system is under heavy load.

By default, StorageGRID attempts to retrieve usage information using strong-global consistency. If strong-global consistency cannot be achieved, StorageGRID attempts to retrieve the usage information at a strong-site consistency.

You must have the `s3:ListAllMyBuckets` permission, or be account root, to complete this operation.

Request example

```
GET /?x-ntap-sg-usage HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization string</em>
Host: <em>host</em>
```

Response example

This example shows an account that has four objects and 12 bytes of data in two buckets. Each bucket contains two objects and six bytes of data.

```
HTTP/1.1 200 OK
Date: Sat, 29 Nov 2015 00:49:05 GMT
Connection: KEEP-ALIVE
Server: StorageGRID/10.2.0
x-amz-request-id: 727237123
Content-Length: 427
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<UsageResult xmlns="http://s3.storagegrid.com/doc/2015-02-01">
<CalculationTime>2014-11-19T05:30:11.000000Z</CalculationTime>
<ObjectCount>4</ObjectCount>
<DataBytes>12</DataBytes>
<Buckets>
<Bucket>
<Name>bucket1</Name>
<ObjectCount>2</ObjectCount>
<DataBytes>6</DataBytes>
</Bucket>
<Bucket>
<Name>bucket2</Name>
<ObjectCount>2</ObjectCount>
<DataBytes>6</DataBytes>
</Bucket>
</Buckets>
</UsageResult>
```

Versioning

Every object version stored will contribute to the `ObjectCount` and `DataBytes` values in the response. Delete markers are not added to the `ObjectCount` total.

Related information

[Consistency controls](#)

Deprecated bucket requests for legacy Compliance

You might need to use the StorageGRID S3 REST API to manage buckets that were created using the legacy Compliance feature.

Compliance feature deprecated

The StorageGRID Compliance feature that was available in previous StorageGRID versions is deprecated and has been replaced by S3 Object Lock.

If you previously enabled the global Compliance setting, the global S3 Object Lock setting is enabled automatically when you upgrade to StorageGRID 11.5. You can no longer create new buckets with Compliance enabled; however, as required, you can use the StorageGRID S3 REST API to manage any existing legacy

Compliant buckets.

[Using S3 Object Lock](#)

[Manage objects with ILM](#)

[NetApp Knowledge Base: How to manage legacy Compliant buckets in StorageGRID 11.5](#)

Deprecated: PUT Bucket request modifications for compliance

The SGCompliance XML element is deprecated. Previously, you could include this StorageGRID custom element in the optional XML request body of PUT Bucket requests to create a Compliant bucket.



The StorageGRID Compliance feature that was available in previous StorageGRID versions is deprecated and has been replaced by S3 Object Lock.

[Using S3 Object Lock](#)

[Manage objects with ILM](#)

[NetApp Knowledge Base: How to manage legacy Compliant buckets in StorageGRID 11.5](#)

You can no longer create new buckets with Compliance enabled. The following error message is returned if you attempt to use the PUT Bucket request modifications for compliance to create a new Compliant bucket:

```
The Compliance feature is deprecated.  
Contact your StorageGRID administrator if you need to create new Compliant  
buckets.
```

Related information

[Manage objects with ILM](#)

[Use a tenant account](#)

Deprecated: GET Bucket compliance request

The GET Bucket compliance request is deprecated. However, you can continue to use this request to determine the compliance settings currently in effect for an existing legacy Compliant bucket.



The StorageGRID Compliance feature that was available in previous StorageGRID versions is deprecated and has been replaced by S3 Object Lock.

[Using S3 Object Lock](#)

[Manage objects with ILM](#)

[NetApp Knowledge Base: How to manage legacy Compliant buckets in StorageGRID 11.5](#)

You must have the s3:GetBucketCompliance permission, or be account root, to complete this operation.

Request example

This example request allows you to determine the compliance settings for the bucket named `mybucket`.

```
GET /mybucket/?x-ntap-sg-compliance HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization string</em>
Host: <em>host</em>
```

Response example

In the response XML, `<SGCompliance>` lists the compliance settings in effect for the bucket. This example response shows the compliance settings for a bucket in which each object will be retained for one year (525,600 minutes), starting from when the object is ingested into the grid. There is currently no legal hold on this bucket. Each object will be automatically deleted after one year.

```
HTTP/1.1 200 OK
Date: <em>date</em>
Connection: <em>connection</em>
Server: StorageGRID/11.1.0
x-amz-request-id: <em>request ID</em>
Content-Length: <em>length</em>
Content-Type: application/xml

<SGCompliance>
  <RetentionPeriodMinutes>525600</RetentionPeriodMinutes>
  <LegalHold>false</LegalHold>
  <AutoDelete>true</AutoDelete>
</SGCompliance>
```

Name	Description
RetentionPeriodMinutes	The length of the retention period for objects added to this bucket, in minutes. The retention period starts when the object is ingested into the grid.
LegalHold	<ul style="list-style-type: none">• True: This bucket is currently under a legal hold. Objects in this bucket cannot be deleted until the legal hold is lifted, even if their retention period has expired.• False: This bucket is not currently under a legal hold. Objects in this bucket can be deleted when their retention period expires.

Name	Description
AutoDelete	<ul style="list-style-type: none"> • True: The objects in this bucket will be deleted automatically when their retention period expires, unless the bucket is under a legal hold. • False: The objects in this bucket will not be deleted automatically when the retention period expires. You must delete these objects manually if you need to delete them.

Error responses

If the bucket was not created to be compliant, the HTTP status code for the response is 404 Not Found, with an S3 error code of XNoSuchBucketCompliance.

Related information

[Manage objects with ILM](#)

[Use a tenant account](#)

Deprecated: PUT Bucket compliance request

The PUT Bucket compliance request is deprecated. However, you can continue to use this request to modify the compliance settings for an existing legacy Compliant bucket. For example, you can place an existing bucket on legal hold or increase its retention period.



The StorageGRID Compliance feature that was available in previous StorageGRID versions is deprecated and has been replaced by S3 Object Lock.

[Using S3 Object Lock](#)

[Manage objects with ILM](#)

[NetApp Knowledge Base: How to manage legacy Compliant buckets in StorageGRID 11.5](#)

You must have the s3:PutBucketCompliance permission, or be account root, to complete this operation.

You must specify a value for every field of the compliance settings when issuing a PUT Bucket compliance request.

Request example

This example request modifies the compliance settings for the bucket named `mybucket`. In this example, objects in `mybucket` will now be retained for two years (1,051,200 minutes) instead of one year, starting from when the object is ingested into the grid. There is no legal hold on this bucket. Each object will be automatically deleted after two years.

```

PUT /mybucket/?x-ntap-sg-compliance HTTP/1.1
Date: <em>date</em>
Authorization: <em>authorization name</em>
Host: <em>host</em>
Content-Length: 152

<SGCompliance>
  <RetentionPeriodMinutes>1051200</RetentionPeriodMinutes>
  <LegalHold>false</LegalHold>
  <AutoDelete>true</AutoDelete>
</SGCompliance>

```

Name	Description
RetentionPeriodMinutes	<p>The length of the retention period for objects added to this bucket, in minutes. The retention period starts when the object is ingested into the grid.</p> <p>Attention: When specifying a new value for RetentionPeriodMinutes, you must specify a value that is equal to or greater than the bucket's current retention period. After the bucket's retention period is set, you cannot decrease that value; you can only increase it.</p>
LegalHold	<ul style="list-style-type: none"> • True: This bucket is currently under a legal hold. Objects in this bucket cannot be deleted until the legal hold is lifted, even if their retention period has expired. • False: This bucket is not currently under a legal hold. Objects in this bucket can be deleted when their retention period expires.
AutoDelete	<ul style="list-style-type: none"> • True: The objects in this bucket will be deleted automatically when their retention period expires, unless the bucket is under a legal hold. • False: The objects in this bucket will not be deleted automatically when the retention period expires. You must delete these objects manually if you need to delete them.

Consistency level for compliance settings

When you update the compliance settings for an S3 bucket with a PUT Bucket compliance request, StorageGRID attempts to update the bucket's metadata across the grid. By default, StorageGRID uses the **strong-global** consistency level to guarantee that all data center sites and all Storage Nodes that contain bucket metadata have read-after-write consistency for the changed compliance settings.

If StorageGRID cannot achieve the **strong-global** consistency level because a data center site or multiple Storage Nodes at a site are unavailable, the HTTP status code for the response is 503 `Service Unavailable`.

If you receive this response, you must contact the grid administrator to ensure that the required storage services are made available as soon as possible. If the grid administrator is unable to make enough of the Storage Nodes at each site available, technical support might direct you to retry the failed request by forcing the **strong-site** consistency level.



Never force the **strong-site** consistency level for PUT bucket compliance unless you have been directed to do so by technical support and unless you understand the potential consequences of using this level.

When the consistency level is reduced to **strong-site**, StorageGRID guarantees that updated compliance settings will have read-after-write consistency only for client requests within a site. This means that the StorageGRID system might temporarily have multiple, inconsistent settings for this bucket until all sites and Storage Nodes are available. The inconsistent settings can result in unexpected and undesired behavior. For example, if you are placing a bucket under a legal hold and you force a lower consistency level, the bucket's previous compliance settings (that is, legal hold off) might continue to be in effect at some data center sites. As a result, objects that you think are on legal hold might be deleted when their retention period expires, either by the user or by AutoDelete, if enabled.

To force the use of the **strong-site** consistency level, reissue the PUT Bucket compliance request and include the `Consistency-Control` HTTP request header, as follows:

```
PUT /mybucket/?x-ntap-sg-compliance HTTP/1.1
Consistency-Control: strong-site
```

Error responses

- If the bucket was not created to be compliant, the HTTP status code for the response is 404 `Not Found`.
- If `RetentionPeriodMinutes` in the request is less than the bucket's current retention period, the HTTP status code is 400 `Bad Request`.

Related information

[Deprecated: PUT Bucket request modifications for compliance](#)

[Use a tenant account](#)

[Manage objects with ILM](#)

Bucket and group access policies

StorageGRID uses the Amazon Web Services (AWS) policy language to allow S3 tenants to control access to buckets and objects within those buckets. The StorageGRID system implements a subset of the S3 REST API policy language. Access policies for the S3 API are written in JSON.

Access policy overview

There are two kinds of access policies supported by StorageGRID.

- **Bucket policies**, which are configured using the GET Bucket policy, PUT Bucket policy, and DELETE Bucket policy S3 API operations. Bucket policies are attached to buckets, so they are configured to control access by users in the bucket owner account or other accounts to the bucket and the objects in it. A bucket policy applies to only one bucket and possibly multiple groups.
- **Group policies**, which are configured using the Tenant Manager or Tenant Management API. Group policies are attached to a group in the account, so they are configured to allow that group to access specific resources owned by that account. A group policy applies to only one group and possibly multiple buckets.

StorageGRID bucket and group policies follow a specific grammar defined by Amazon. Inside each policy is an array of policy statements, and each statement contains the following elements:

- Statement ID (Sid) (optional)
- Effect
- Principal/NotPrincipal
- Resource/NotResource
- Action/NotAction
- Condition (optional)

Policy statements are built using this structure to specify permissions: Grant <Effect> to allow/deny <Principal> to perform <Action> on <Resource> when <Condition> applies.

Each policy element is used for a specific function:

Element	Description
Sid	The Sid element is optional. The Sid is only intended as a description for the user. It is stored but not interpreted by the StorageGRID system.
Effect	Use the Effect element to establish whether the specified operations are allowed or denied. You must identify operations you allow (or deny) on buckets or objects using the supported Action element keywords.
Principal/NotPrincipal	<p>You can allow users, groups, and accounts to access specific resources and perform specific actions. If no S3 signature is included in the request, anonymous access is allowed by specifying the wildcard character (*) as the principal. By default, only the account root has access to resources owned by the account.</p> <p>You only need to specify the Principal element in a bucket policy. For group policies, the group to which the policy is attached is the implicit Principal element.</p>

Element	Description
Resource/NotResource	The Resource element identifies buckets and objects. You can allow or deny permissions to buckets and objects using the Amazon Resource Name (ARN) to identify the resource.
Action/NotAction	The Action and Effect elements are the two components of permissions. When a group requests a resource, they are either granted or denied access to the resource. Access is denied unless you specifically assign permissions, but you can use explicit deny to override a permission granted by another policy.
Condition	The Condition element is optional. Conditions allow you to build expressions to determine when a policy should be applied.

In the Action element, you can use the wildcard character (*) to specify all operations, or a subset of operations. For example, this Action matches permissions such as `s3:GetObject`, `s3:PutObject`, and `s3:DeleteObject`.

```
s3:*Object
```

In the Resource element, you can use the wildcard characters (*) and (?). While the asterisk (*) matches 0 or more characters, the question mark (?) matches any single character.

In the Principal element, wildcard characters are not supported except to set anonymous access, which grants permission to everyone. For example, you set the wildcard (*) as the Principal value.

```
"Principal": "*"

```

In the following example, the statement is using the Effect, Principal, Action, and Resource elements. This example shows a complete bucket policy statement that uses the Effect "Allow" to give the Principals, the admin group `federated-group/admin` and the finance group `federated-group/finance`, permissions to perform the Action `s3:ListBucket` on the bucket named `mybucket` and the Action `s3:GetObject` on all objects inside that bucket.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::27233906934684427525:federated-group/admin",
          "arn:aws:iam::27233906934684427525:federated-group/finance"
        ]
      },
      "Action": [
        "s3:ListBucket",
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:iam:s3::mybucket",
        "arn:aws:iam:s3::mybucket/*"
      ]
    }
  ]
}
```

The bucket policy has a size limit of 20,480 bytes, and the group policy has a size limit of 5,120 bytes.

Related information

[Use a tenant account](#)

Consistency control settings for policies

By default, any updates you make to group policies are eventually consistent. Once a group policy becomes consistent, the changes can take an additional 15 minutes to take effect, because of policy caching. By default, any updates you make to bucket policies are also eventually consistent.

As required, you can change the consistency guarantees for bucket policy updates. For example, you might want a change to a bucket policy to become effective as soon as possible for security reasons.

In this case, you can either set the `Consistency-Control` header in the PUT Bucket policy request, or you can use the PUT Bucket consistency request. When changing the consistency control for this request, you must use the value **all**, which provides the highest guarantee of read-after-write consistency. If you specify any other consistency control value in a header for the PUT Bucket consistency request, the request will be rejected. If you specify any other value for a PUT Bucket policy request, the value will be ignored. Once a bucket policy becomes consistent, the changes can take an additional 8 seconds to take effect, because of policy caching.



If you set the consistency level to **all** to force a new bucket policy to become effective sooner, be sure to set the bucket-level control back to its original value when you are done. Otherwise, all future bucket requests will use the **all** setting.

Using the ARN in policy statements

In policy statements, the ARN is used in Principal and Resource elements.

- Use this syntax to specify the S3 resource ARN:

```
arn:aws:s3:::bucket-name
arn:aws:s3:::bucket-name/object_key
```

- Use this syntax to specify the identity resource ARN (users and groups):

```
arn:aws:iam::account_id:root
arn:aws:iam::account_id:user/user_name
arn:aws:iam::account_id:group/group_name
arn:aws:iam::account_id:federated-user/user_name
arn:aws:iam::account_id:federated-group/group_name
```

Other considerations:

- You can use the asterisk (*) as a wildcard to match zero or more characters inside the object key.
- International characters, which can be specified in the object key, should be encoded using JSON UTF-8 or using JSON \u escape sequences. Percent-encoding is not supported.

[RFC 2141 URN Syntax](#)

The HTTP request body for the PUT Bucket policy operation must be encoded with charset=UTF-8.

Specifying resources in a policy

In policy statements, you can use the Resource element to specify the bucket or object for which permissions are allowed or denied.

- Each policy statement requires a Resource element. In a policy, resources are denoted by the element `Resource`, or alternatively, `NotResource` for exclusion.
- You specify resources with an S3 resource ARN. For example:

```
"Resource": "arn:aws:s3:::mybucket/*"
```

- You can also use policy variables inside the object key. For example:

```
"Resource": "arn:aws:s3:::mybucket/home/${aws:username}/*"
```

- The resource value can specify a bucket that does not yet exist when a group policy is created.

Related information

Specifying principals in a policy

Use the Principal element to identify the user, group, or tenant account that is allowed/denied access to the resource by the policy statement.

- Each policy statement in a bucket policy must include a Principal element. Policy statements in a group policy do not need the Principal element because the group is understood to be the principal.
- In a policy, principals are denoted by the element “Principal,” or alternatively “NotPrincipal” for exclusion.
- Account-based identities must be specified using an ID or an ARN:

```
"Principal": { "AWS": "account_id"}  
"Principal": { "AWS": "identity_arn" }
```

- This example uses the tenant account ID 27233906934684427525, which includes the account root and all users in the account:

```
"Principal": { "AWS": "27233906934684427525" }
```

- You can specify just the account root:

```
"Principal": { "AWS": "arn:aws:iam::27233906934684427525:root" }
```

- You can specify a specific federated user ("Alex"):

```
"Principal": { "AWS": "arn:aws:iam::27233906934684427525:federated-  
user/Alex" }
```

- You can specify a specific federated group ("Managers"):

```
"Principal": { "AWS": "arn:aws:iam::27233906934684427525:federated-  
group/Managers" }
```

- You can specify an anonymous principal:

```
"Principal": ""
```

- To avoid ambiguity, you can use the user UUID instead of the username:

```
arn:aws:iam::27233906934684427525:user-uuid/de305d54-75b4-431b-adb2-eb6b9e546013
```

For example, suppose Alex leaves the organization and the username `Alex` is deleted. If a new Alex joins the organization and is assigned the same `Alex` username, the new user might unintentionally inherit the permissions granted to the original user.

- The principal value can specify a group/user name that does not yet exist when a bucket policy is created.

Specifying permissions in a policy

In a policy, the Action element is used to allow/deny permissions to a resource. There are a set of permissions that you can specify in a policy, which are denoted by the element "Action," or alternatively, "NotAction" for exclusion. Each of these elements maps to specific S3 REST API operations.

The tables lists the permissions that apply to buckets and the permissions that apply to objects.



Amazon S3 now uses the `s3:PutReplicationConfiguration` permission for both the PUT and DELETE Bucket replication actions. StorageGRID uses separate permissions for each action, which matches the original Amazon S3 specification.



A DELETE is performed when a PUT is used to overwrite an existing value.

Permissions that apply to buckets

Permissions	S3 REST API operations	Custom for StorageGRID
<code>s3:CreateBucket</code>	PUT Bucket	
<code>s3:DeleteBucket</code>	DELETE Bucket	
<code>s3:DeleteBucketMetadataNotification</code>	DELETE Bucket metadata notification configuration	Yes
<code>s3:DeleteBucketPolicy</code>	DELETE Bucket policy	
<code>s3:DeleteReplicationConfiguration</code>	DELETE Bucket replication	Yes, separate permissions for PUT and DELETE*
<code>s3:GetBucketAcl</code>	GET Bucket ACL	
<code>s3:GetBucketCompliance</code>	GET Bucket compliance (deprecated)	Yes
<code>s3:GetBucketConsistency</code>	GET Bucket consistency	Yes
<code>s3:GetBucketCORS</code>	GET Bucket cors	

Permissions	S3 REST API operations	Custom for StorageGRID
s3:GetEncryptionConfiguration	GET Bucket encryption	
s3:GetBucketLastAccessTime	GET Bucket last access time	Yes
s3:GetBucketLocation	GET Bucket location	
s3:GetBucketMetadataNotification	GET Bucket metadata notification configuration	Yes
s3:GetBucketNotification	GET Bucket notification	
s3:GetBucketObjectLockConfiguration	GET Object Lock Configuration	
s3:GetBucketPolicy	GET Bucket policy	
s3:GetBucketTagging	GET Bucket tagging	
s3:GetBucketVersioning	GET Bucket versioning	
s3:GetLifecycleConfiguration	GET Bucket lifecycle	
s3:GetReplicationConfiguration	GET Bucket replication	
s3:ListAllMyBuckets	<ul style="list-style-type: none"> • GET Service • GET Storage Usage 	Yes, for GET Storage Usage
s3:ListBucket	<ul style="list-style-type: none"> • GET Bucket (List Objects) • HEAD Bucket • POST Object restore 	
s3:ListBucketMultipartUploads	<ul style="list-style-type: none"> • List Multipart Uploads • POST Object restore 	
s3:ListBucketVersions	GET Bucket versions	
s3:PutBucketCompliance	PUT Bucket compliance (deprecated)	Yes
s3:PutBucketConsistency	PUT Bucket consistency	Yes

Permissions	S3 REST API operations	Custom for StorageGRID
s3:PutBucketCORS	<ul style="list-style-type: none"> • DELETE Bucket cors† • PUT Bucket cors 	
s3:PutEncryptionConfiguration	<ul style="list-style-type: none"> • DELETE Bucket encryption • PUT Bucket encryption 	
s3:PutBucketLastAccessTime	PUT Bucket last access time	Yes
s3:PutBucketMetadataNotification	PUT Bucket metadata notification configuration	Yes
s3:PutBucketNotification	PUT Bucket notification	
s3:PutBucketObjectLockConfiguration	PUT Bucket with the x-amz-bucket-object-lock-enabled: true request header (also requires the s3:CreateBucket permission)	
s3:PutBucketPolicy	PUT Bucket policy	
s3:PutBucketTagging	<ul style="list-style-type: none"> • DELETE Bucket tagging† • PUT Bucket tagging 	
s3:PutBucketVersioning	PUT Bucket versioning	
s3:PutLifecycleConfiguration	<ul style="list-style-type: none"> • DELETE Bucket lifecycle† • PUT Bucket lifecycle 	
s3:PutReplicationConfiguration	PUT Bucket replication	Yes, separate permissions for PUT and DELETE*

Permissions that apply to objects

Permissions	S3 REST API operations	Custom for StorageGRID
s3:AbortMultipartUpload	<ul style="list-style-type: none"> • Abort Multipart Upload • POST Object restore 	
s3:DeleteObject	<ul style="list-style-type: none"> • DELETE Object • DELETE Multiple Objects • POST Object restore 	

Permissions	S3 REST API operations	Custom for StorageGRID
s3:DeleteObjectTagging	DELETE Object Tagging	
s3:DeleteObjectVersionTagging	DELETE Object Tagging (a specific version of the object)	
s3:DeleteObjectVersion	DELETE Object (a specific version of the object)	
s3:GetObject	<ul style="list-style-type: none"> • GET Object • HEAD Object • POST Object restore 	
s3:GetObjectAcl	GET Object ACL	
s3:GetObjectLegalHold	GET Object legal hold	
s3:GetObjectRetention	GET Object retention	
s3:GetObjectTagging	GET Object Tagging	
s3:GetObjectVersionTagging	GET Object Tagging (a specific version of the object)	
s3:GetObjectVersion	GET Object (a specific version of the object)	
s3:ListMultipartUploadParts	List Parts, POST Object restore	
s3:PutObject	<ul style="list-style-type: none"> • PUT Object • PUT Object - Copy • POST Object restore • Initiate Multipart Upload • Complete Multipart Upload • Upload Part • Upload Part - Copy 	
s3:PutObjectLegalHold	PUT Object legal hold	
s3:PutObjectRetention	PUT Object retention	
s3:PutObjectTagging	PUT Object Tagging	

Permissions	S3 REST API operations	Custom for StorageGRID
s3:PutObjectVersionTagging	PUT Object Tagging (a specific version of the object)	
s3:PutOverwriteObject	<ul style="list-style-type: none"> • PUT Object • PUT Object - Copy • PUT Object tagging • DELETE Object tagging • Complete Multipart Upload 	Yes
s3:RestoreObject	POST Object restore	

Using the PutOverwriteObject permission

The s3:PutOverwriteObject permission is a custom StorageGRID permission that applies to operations that create or update objects. The setting of this permission determines whether the client can overwrite an object's data, user-defined metadata, or S3 object tagging.

Possible settings for this permission include:

- **Allow:** The client can overwrite an object. This is the default setting.
- **Deny:** The client cannot overwrite an object. When set to Deny, the PutOverwriteObject permission works as follows:
 - If an existing object is found at the same path:
 - The object's data, user-defined metadata, or S3 object tagging cannot be overwritten.
 - Any ingest operations in progress are cancelled, and an error is returned.
 - If S3 versioning is enabled, the Deny setting prevents PUT Object tagging or DELETE Object tagging operations from modifying the TagSet for an object and its noncurrent versions.
 - If an existing object is not found, this permission has no effect.
- When this permission is not present, the effect is the same as if Allow were set.



If the current S3 policy allows overwrite, and the PutOverwriteObject permission is set to Deny, the client cannot overwrite an object's data, user-defined metadata, or object tagging. In addition, if the **Prevent Client Modification** check box is selected (**Configuration > Grid Options**), that setting overrides the setting of the PutOverwriteObject permission.

Related information

[S3 group policy examples](#)

Specifying conditions in a policy

Conditions define when a policy will be in effect. Conditions consist of operators and key-value pairs.

Conditions use key-value pairs for evaluation. A Condition element can contain multiple conditions, and each condition can contain multiple key-value pairs. The condition block uses the following format:

```
Condition: {
  <em>condition_type</em>: {
    <em>condition_key</em>: <em>condition_values</em>
```

In the following example, the `IpAddress` condition uses the `SourceIp` condition key.

```
"Condition": {
  "IpAddress": {
    "aws:SourceIp": "54.240.143.0/24"
    ...
  },
  ...
```

Supported condition operators

Condition operators are categorized as follows:

- String
- Numeric
- Boolean
- IP address
- Null check

Condition operators	Description
StringEquals	Compares a key to a string value based on exact matching (case sensitive).
StringNotEquals	Compares a key to a string value based on negated matching (case sensitive).
StringEqualsIgnoreCase	Compares a key to a string value based on exact matching (ignores case).
StringNotEqualsIgnoreCase	Compares a key to a string value based on negated matching (ignores case).
StringLike	Compares a key to a string value based on exact matching (case sensitive). Can include * and ? wildcard characters.
StringNotLike	Compares a key to a string value based on negated matching (case sensitive). Can include * and ? wildcard characters.

Condition operators	Description
NumericEquals	Compares a key to a numeric value based on exact matching.
NumericNotEquals	Compares a key to a numeric value based on negated matching.
NumericGreaterThan	Compares a key to a numeric value based on “greater than” matching.
NumericGreaterThanEquals	Compares a key to a numeric value based on “greater than or equals” matching.
NumericLessThan	Compares a key to a numeric value based on “less than” matching.
NumericLessThanEquals	Compares a key to a numeric value based on “less than or equals” matching.
Bool	Compares a key to a Boolean value based on “true or false” matching.
IpAddress	Compares a key to an IP address or range of IP addresses.
NotIpAddress	Compares a key to an IP address or range of IP addresses based on negated matching.
Null	Checks if a condition key is present in the current request context.

Supported condition keys

Category	Applicable condition keys	Description
IP operators	aws:SourceIp	<p>Will compare to the IP address from which the request was sent. Can be used for bucket or object operations.</p> <p>Note: If the S3 request was sent through the Load Balancer service on Admin Nodes and Gateways Nodes, this will compare to the IP address upstream of the Load Balancer service.</p> <p>Note: If a third-party, non-transparent load balancer is used, this will compare to the IP address of that load balancer. Any X-Forwarded-For header will be ignored since its validity cannot be ascertained.</p>
Resource/Identity	aws:username	Will compare to the sender's username from which the request was sent. Can be used for bucket or object operations.
S3:ListBucket and S3:ListBucketVersions permissions	s3:delimiter	Will compare to the delimiter parameter specified in a GET Bucket or GET Bucket Object versions request.
S3:ListBucket and S3:ListBucketVersions permissions	s3:max-keys	Will compare to the max-keys parameter specified in a GET Bucket or GET Bucket Object versions request.
S3:ListBucket and S3:ListBucketVersions permissions	s3:prefix	Will compare to the prefix parameter specified in a GET Bucket or GET Bucket Object versions request.

Specifying variables in a policy

You can use variables in policies to populate policy information when it is available. You can use policy variables in the `Resource` element and in string comparisons in the `Condition` element.

In this example, the variable `${aws:username}` is part of the `Resource` element:

```
"Resource": "arn:aws:s3:::_bucket-name/home_/${aws:username}/*"
```

In this example, the variable `${aws:username}` is part of the condition value in the condition block:

```
"Condition": {
  "StringLike": {
    "s3:prefix": "${aws:username}/*"
    ...
  },
  ...
}
```

Variable	Description
<code>\${aws:SourceIp}</code>	Uses the SourceIp key as the provided variable.
<code>\${aws:username}</code>	Uses the username key as the provided variable.
<code>\${s3:prefix}</code>	Uses the service-specific prefix key as the provided variable.
<code>\${s3:max-keys}</code>	Uses the service-specific max-keys key as the provided variable.
<code>\${*}</code>	Special character. Uses the character as a literal * character.
<code>\${?}</code>	Special character. Uses the character as a literal ? character.
<code>\${\$}</code>	Special character. Uses the character as a literal \$ character.

Creating policies requiring special handling

Sometimes a policy can grant permissions that are dangerous for security or dangerous for continued operations, such as locking out the root user of the account. The StorageGRID S3 REST API implementation is less restrictive during policy validation than Amazon, but equally strict during policy evaluation.

Policy description	Policy type	Amazon behavior	StorageGRID behavior
Deny self any permissions to the root account	Bucket	Valid and enforced, but root user account retains permission for all S3 bucket policy operations	Same
Deny self any permissions to user/group	Group	Valid and enforced	Same

Policy description	Policy type	Amazon behavior	StorageGRID behavior
Allow a foreign account group any permission	Bucket	Invalid principal	Valid, but permissions for all S3 bucket policy operations return a 405 Method Not Allowed error when allowed by a policy
Allow a foreign account root or user any permission	Bucket	Valid, but permissions for all S3 bucket policy operations return a 405 Method Not Allowed error when allowed by a policy	Same
Allow everyone permissions to all actions	Bucket	Valid, but permissions for all S3 bucket policy operations return a 405 Method Not Allowed error for the foreign account root and users	Same
Deny everyone permissions to all actions	Bucket	Valid and enforced, but root user account retains permission for all S3 bucket policy operations	Same
Principal is a non-existent user or group	Bucket	Invalid principal	Valid
Resource is a non-existent S3 bucket	Group	Valid	Same
Principal is a local group	Bucket	Invalid principal	Valid
Policy grants a non-owner account (including anonymous accounts) permissions to PUT objects	Bucket	Valid. Objects are owned by the creator account, and the bucket policy does not apply. The creator account must grant access permissions for the object using object ACLs.	Valid. Objects are owned by the bucket owner account. Bucket policy applies.

Write-once-read-many (WORM) protection

You can create write-once-read-many (WORM) buckets to protect data, user-defined object metadata, and S3 object tagging. You configure the WORM buckets to allow the creation of new objects and to prevent overwrites or deletion of existing content. Use one of the approaches described here.

To ensure that overwrites are always denied, you can:

- From the Grid Manager, go to **Configuration > Grid Options**, and select the **Prevent Client Modification** check box.
- Apply the following rules and S3 policies:
 - Add a PutOverwriteObject DENY operation to the S3 policy.
 - Add a DeleteObject DENY operation to the S3 policy.
 - Add a PUT Object ALLOW operation to the S3 policy.



Setting DeleteObject to DENY in an S3 policy does not prevent ILM from deleting objects when a rule such as “zero copies after 30 days” exists.



Even when all of these rules and policies are applied, they do not guard against concurrent writes (see Situation A). They do guard against sequential completed overwrites (see Situation B).

Situation A: Concurrent writes (not guarded against)

```
/mybucket/important.doc
PUT#1 ---> OK
PUT#2 -----> OK
```

Situation B: Sequential completed overwrites (guarded against)

```
/mybucket/important.doc
PUT#1 -----> PUT#2 ---X (denied)
```

Related information

[Manage objects with ILM](#)

[Creating policies requiring special handling](#)

[How StorageGRID ILM rules manage objects](#)

[S3 group policy examples](#)

S3 policy examples

Use the examples in this section to build StorageGRID access policies for buckets and groups.

S3 bucket policy examples

Bucket policies specify the access permissions for the bucket that the policy is attached to. Bucket policies are configured using the S3 PutBucketPolicy API.

A bucket policy can be configured using the AWS CLI as per the following command:

```
> aws s3api put-bucket-policy --bucket examplebucket --policy
<em>file://policy.json</em>
```

Example: Allow everyone read-only access to a bucket

In this example, everyone, including anonymous, is allowed to list objects in the bucket and perform Get Object operations on all objects in the bucket. All other operations will be denied. Note that this policy might not be particularly useful since no one except the account root has permissions to write to the bucket.

```
{
  "Statement": [
    {
      "Sid": "AllowEveryoneReadOnlyAccess",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [ "s3:GetObject", "s3:ListBucket" ],
      "Resource":
[ "arn:aws:s3:::examplebucket", "arn:aws:s3:::examplebucket/*" ]
    }
  ]
}
```

Example: Allow everyone in one account full access, and everyone in another account read-only access to a bucket

In this example, everyone in one specified account is allowed full access to a bucket, while everyone in another specified account is only permitted to List the bucket and perform GetObject operations on objects in the bucket beginning with the `shared/` object key prefix.



In StorageGRID, objects created by a non-owner account (including anonymous accounts) are owned by the bucket owner account. The bucket policy applies to these objects.


```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "95390887230002558202"
      },
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::examplebucket",
        "arn:aws:s3:::examplebucket/*"
      ]
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "31181711887329436680"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::examplebucket/shared/*"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "31181711887329436680"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::examplebucket",
      "Condition": {
        "StringLike": {
          "s3:prefix": "shared/*"
        }
      }
    }
  ]
}

```

Example: Allow everyone read-only access to a bucket and full access by specified group

In this example, everyone including anonymous, is allowed to List the bucket and perform GET Object operations on all objects in the bucket, while only users belonging the group `Marketing` in the specified account are allowed full access.

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::95390887230002558202:federated-
group/Marketing"
      },
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::examplebucket",
        "arn:aws:s3:::examplebucket/*"
      ]
    },
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": ["s3:ListBucket", "s3:GetObject"],
      "Resource": [
        "arn:aws:s3:::examplebucket",
        "arn:aws:s3:::examplebucket/*"
      ]
    }
  ]
}

```

Example: Allow everyone read and write access to a bucket if client in IP range

In this example, everyone, including anonymous, is allowed to List the bucket and perform any Object operations on all objects in the bucket, provided that the requests come from a specified IP range (54.240.143.0 to 54.240.143.255, except 54.240.143.188). All other operations will be denied, and all requests outside of the IP range will be denied.

```

{
  "Statement": [
    {
      "Sid": "AllowEveryoneReadWriteAccessIfInSourceIpRange",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [ "s3:*Object", "s3:ListBucket" ],
      "Resource":
[ "arn:aws:s3:::examplebucket", "arn:aws:s3:::examplebucket/*" ],
      "Condition": {
        "IpAddress": { "aws:SourceIp": "54.240.143.0/24" },
        "NotIpAddress": { "aws:SourceIp": "54.240.143.188" }
      }
    }
  ]
}

```

Example: Allow full access to a bucket exclusively by a specified federated user

In this example, the federated user Alex is allowed full access to the `examplebucket` bucket and its objects. All other users, including 'root', are explicitly denied all operations. Note however that 'root' is never denied permissions to Put/Get/DeleteBucketPolicy.

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::95390887230002558202:federated-user/Alex"
      },
      "Action": [
        "s3:*"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket",
        "arn:aws:s3:::examplebucket/*"
      ]
    },
    {
      "Effect": "Deny",
      "NotPrincipal": {
        "AWS": "arn:aws:iam::95390887230002558202:federated-user/Alex"
      },
      "Action": [
        "s3:*"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket",
        "arn:aws:s3:::examplebucket/*"
      ]
    }
  ]
}

```

Example: PutOverwriteObject permission

In this example, the `Deny` Effect for `PutOverwriteObject` and `DeleteObject` ensures that no one can overwrite or delete the object's data, user-defined metadata, and S3 object tagging.

```

{
  "Statement": [
    {
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:PutOverwriteObject",
        "s3:DeleteObject",
        "s3:DeleteObjectVersion"
      ],
      "Resource": "arn:aws:s3:::wormbucket/*"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::95390887230002558202:federated-
group/SomeGroup"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::wormbucket"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::95390887230002558202:federated-
group/SomeGroup"
      },
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::wormbucket/*"
    }
  ]
}

```

Related information

[Operations on buckets](#)

S3 group policy examples

Group policies specify the access permissions for the group that the policy is attached to. There is no Principal element in the policy since it is implicit. Group policies are configured using the Tenant Manager or the API.

Example: Setting the group policy using the Tenant Manager

When using the Tenant Manager to add or edit a group, you can select how you want to create the group policy that defines which S3 access permissions members of this group will have, as follows:

- **No S3 Access:** Default option. Users in this group do not have access to S3 resources, unless access is granted with a bucket policy. If you select this option, only the root user will have access to S3 resources by default.
- **Read Only Access:** Users in this group have read-only access to S3 resources. For example, users in this group can list objects and read object data, metadata, and tags. When you select this option, the JSON string for a read-only group policy appears in the text box. You cannot edit this string.
- **Full Access:** Users in this group have full access to S3 resources, including buckets. When you select this option, the JSON string for a full-access group policy appears in the text box. You cannot edit this string.
- **Custom:** Users in the group are granted the permissions you specify in the text box.

In this example, members of the group are only permitted to list and access their specific folder (key prefix) in the specified bucket.

```
{
  "Statement": [
    {
      "Sid": "AllowListBucketOfASpecificUserPrefix",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::department-bucket",
      "Condition": {
        "StringLike": {
          "s3:prefix": "${aws:username}/*"
        }
      }
    },
    {
      "Sid": "AllowUserSpecificActionsOnlyInTheSpecificFolder",
      "Effect": "Allow",
      "Action": "s3:*Object",
      "Resource": "arn:aws:s3:::department-bucket/${aws:username}/*"
    }
  ]
}
```

Example: Allow group full access to all buckets

In this example, all members of the group are permitted full access to all buckets owned by the tenant account unless explicitly denied by bucket policy.

```
{
  "Statement": [
    {
      "Action": "s3:*",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::*"
    }
  ]
}
```

Example: Allow group read-only access to all buckets

In this example, all members of the group have read-only access to S3 resources, unless explicitly denied by the bucket policy. For example, users in this group can list objects and read object data, metadata, and tags.

```
{
  "Statement": [
    {
      "Sid": "AllowGroupReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:ListBucket",
        "s3:ListBucketVersions",
        "s3:GetObject",
        "s3:GetObjectTagging",
        "s3:GetObjectVersion",
        "s3:GetObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::*"
    }
  ]
}
```

Example: Allow group members full access to only their “folder” in a bucket

In this example, members of the group are only permitted to list and access their specific folder (key prefix) in the specified bucket. Note that access permissions from other group policies and the bucket policy should be considered when determining the privacy of these folders.

```
{
  "Statement": [
    {
      "Sid": "AllowListBucketOfASpecificUserPrefix",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::department-bucket",
      "Condition": {
        "StringLike": {
          "s3:prefix": "${aws:username}/*"
        }
      }
    },
    {
      "Sid": "AllowUserSpecificActionsOnlyInTheSpecificUserPrefix",
      "Effect": "Allow",
      "Action": "s3:*Object",
      "Resource": "arn:aws:s3:::department-bucket/${aws:username}/*"
    }
  ]
}
```

Related information

[Use a tenant account](#)

[Using the PutOverwriteObject permission](#)

[Write-once-read-many \(WORM\) protection](#)

Configuring security for the REST API

You should review the security measures implemented for the REST API and understand how to secure your system.

How StorageGRID provides security for the REST API

You should understand how the StorageGRID system implements security, authentication, and authorization for the REST API.

StorageGRID uses the following security measures.

- Client communications with the Load Balancer service use HTTPS if HTTPS is configured for the load balancer endpoint.

When you configure a load balancer endpoint, HTTP can optionally be enabled. For example, you might want to use HTTP for testing or other non-production purposes. See the instructions for administering StorageGRID for more information.

- By default, StorageGRID uses HTTPS for client communications with Storage Nodes and the CLB service on Gateway Nodes.

HTTP can optionally be enabled for these connections. For example, you might want to use HTTP for testing or other non-production purposes. See the instructions for administering StorageGRID for more information.



The CLB service is deprecated.

- Communications between StorageGRID and the client are encrypted using TLS.
- Communications between the Load Balancer service and Storage Nodes within the grid are encrypted whether the load balancer endpoint is configured to accept HTTP or HTTPS connections.
- Clients must supply HTTP authentication headers to StorageGRID to perform REST API operations.

Security certificates and client applications

Clients can connect to the Load Balancer service on Gateway Nodes or Admin Nodes, directly to Storage Nodes, or to the CLB service on Gateway Nodes.

In all cases, client applications can make TLS connections using either a custom server certificate uploaded by the grid administrator or a certificate generated by the StorageGRID system:

- When client applications connect to the Load Balancer service, they do so using the certificate that was configured for the specific load balancer endpoint used to make the connection. Each endpoint has its own certificate, which is either a custom server certificate uploaded by the grid administrator or a certificate that the grid administrator generated in StorageGRID when configuring the endpoint.
- When client applications connect directly to a Storage Node or to the CLB service on Gateway Nodes, they use either the system-generated server certificates that were generated for Storage Nodes when the StorageGRID system was installed (which are signed by the system certificate authority), or a single custom server certificate that is supplied for the grid by a grid administrator.

Clients should be configured to trust the certificate authority that signed whichever certificate they use to establish TLS connections.

See the instructions for administering StorageGRID for information on configuring load balancer endpoints, and for instructions on adding a single custom server certificate for TLS connections directly to Storage Nodes or to the CLB service on Gateway Nodes.

Summary

The following table shows how security issues are implemented in the S3 and Swift REST APIs:

Security issue	Implementation for REST API
Connection security	TLS
Server authentication	X.509 server certificate signed by system CA or custom server certificate supplied by administrator

Security issue	Implementation for REST API
Client authentication	<ul style="list-style-type: none"> • S3: S3 account (access key ID and secret access key) • Swift: Swift account (user name and password)
Client authorization	<ul style="list-style-type: none"> • S3: Bucket ownership and all applicable access control policies • Swift: Administrator role access

Related information

[Administer StorageGRID](#)

Supported hashing and encryption algorithms for TLS libraries

The StorageGRID system supports a limited set of cipher suites that client applications can use when establishing a Transport Layer Security (TLS) session.

Supported versions of TLS

StorageGRID supports TLS 1.2 and TLS 1.3.



SSLv3 and TLS 1.1 (or earlier versions) are no longer supported.

Supported cipher suites

TLS version	IANA name of cipher suite
1.2	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
1.2	TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
1.3	TLS_AES_256_GCM_SHA384
1.3	TLS_CHACHA20_POLY1305_SHA256
1.3	TLS_AES_128_GCM_SHA256

Deprecated cipher suites

The following cipher suites are deprecated. Support for these ciphers will be removed in a future release.

IANA Name
TLS_RSA_WITH_AES_128_GCM_SHA256

IANA Name
TLS_RSA_WITH_AES_256_GCM_SHA384

Related information

[How client connections can be configured](#)

Monitoring and auditing operations

You can monitor workloads and efficiencies for client operations by viewing transaction trends for the entire grid, or for specific nodes. You can use audit messages to monitor client operations and transactions.

- [Monitoring object ingest and retrieval rates](#)
- [Accessing and reviewing audit logs](#)

Monitoring object ingest and retrieval rates

You can monitor object ingest and retrieval rates as well as metrics for object counts, queries, and verification. You can view the number of successful and failed attempts by client applications to read, write, and modify objects in the StorageGRID system.

Steps

1. Sign in to the Grid Manager using a supported browser.
2. On the Dashboard, locate the Protocol Operations section.

This section summarizes the number of client operations performed by your StorageGRID system. Protocol rates are averaged over the last two minutes.

3. Select **Nodes**.
4. From the Nodes home page (deployment level), click the **Load Balancer** tab.

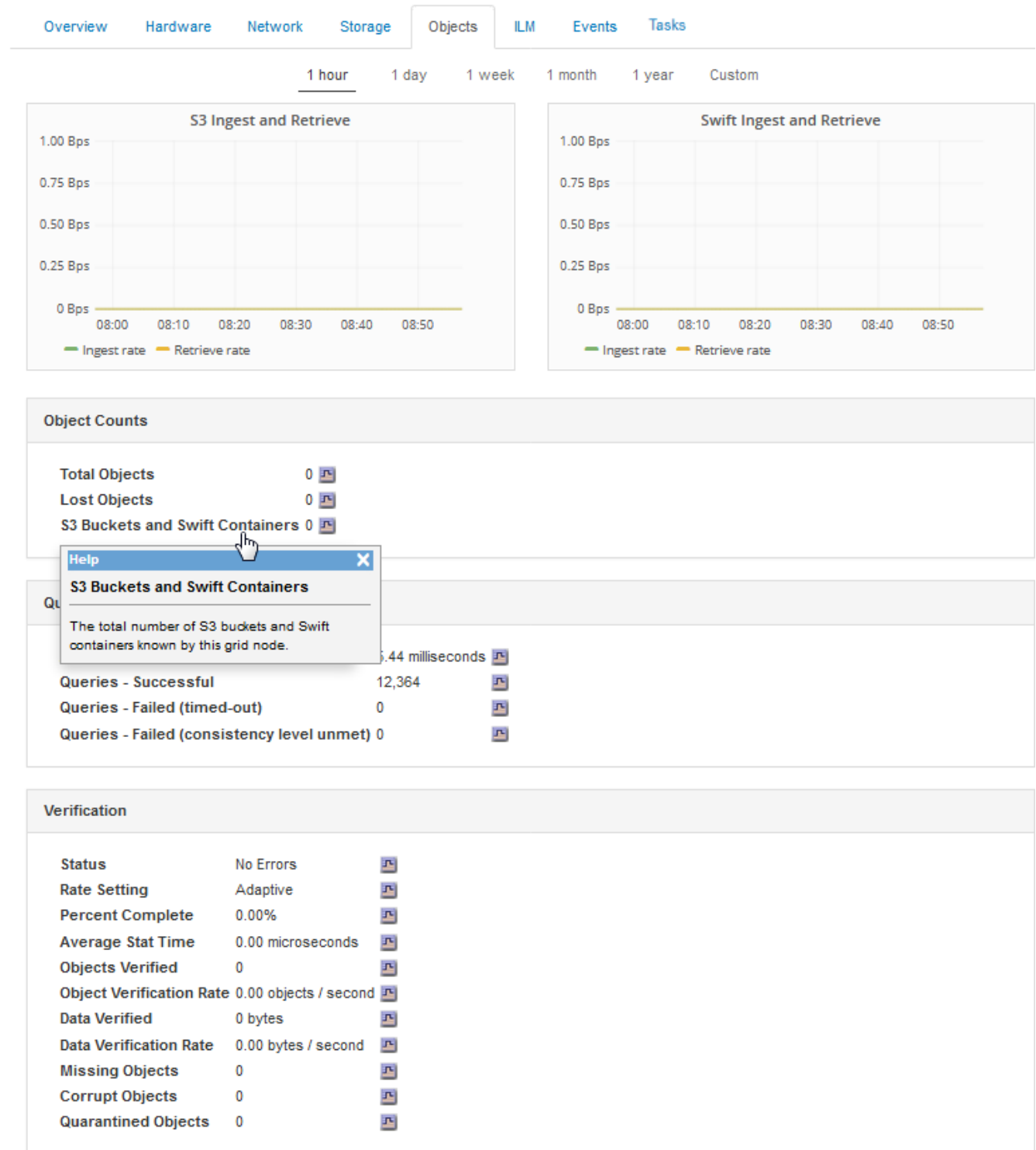
The charts show trends for all client traffic directed to load balancer endpoints within the grid. You can select a time interval in hours, days, weeks, months, or years, or you can apply a custom interval.

5. From the Nodes home page (deployment level), click the **Objects** tab.

The chart shows ingest and retrieve rates for your entire StorageGRID system in bytes per second and total bytes. You can select a time interval in hours, days, weeks, months, or years, or you can apply a custom interval.

6. To see information for a particular Storage Node, select the node from the list on the left, and click the **Objects** tab.

The chart shows the object ingest and retrieval rates for this Storage Node. The tab also includes metrics for object counts, queries, and verification. You can click the labels to see the definitions of these metrics.



7. If you want even more detail:

- Select **Support > Tools > Grid Topology**.
- Select **site > Overview > Main**.

The API Operations section displays summary information for the entire grid.

- Select **Storage Node > LDR > client application > Overview > Main**

The Operations section displays summary information for the selected Storage Node.

Accessing and reviewing audit logs

Audit messages are generated by StorageGRID services and stored in text log files. API-specific audit messages in the audit logs provide critical security, operation, and performance monitoring data that can help you evaluate the health of your system.

What you'll need

- You must have specific access permissions.
- You must have the `Passwords.txt` file.
- You must know the IP address of an Admin Node.

About this task

The active audit log file is named `audit.log`, and it is stored on Admin Nodes.

Once a day, the active `audit.log` file is saved, and a new `audit.log` file is started. The name of the saved file indicates when it was saved, in the format `yyyy-mm-dd.txt`.

After a day, the saved file is compressed and renamed, in the format `yyyy-mm-dd.txt.gz`, which preserves the original date.

This example shows the active `audit.log` file, the previous day's file (`2018-04-15.txt`), and the compressed file for the prior day (`2018-04-14.txt.gz`).

```
audit.log
2018-04-15.txt
2018-04-14.txt.gz
```

Steps

1. Log in to an Admin Node:
 - a. Enter the following command:
`ssh admin@primary_Admin_Node_IP`
 - b. Enter the password listed in the `Passwords.txt` file.
2. Go to the directory containing the audit log files:

```
cd /var/local/audit/export
```

3. View the current or a saved audit log file, as required.

S3 operations tracked in the audit logs

Several bucket operations and object operations are tracked in the StorageGRID audit logs.

Bucket operations tracked in the audit logs

- DELETE Bucket
- DELETE Bucket tagging
- DELETE Multiple Objects
- GET Bucket (List Objects)
- GET Bucket Object versions
- GET Bucket tagging
- HEAD Bucket
- PUT Bucket
- PUT Bucket compliance
- PUT Bucket tagging
- PUT Bucket versioning

Object operations tracked in the audit logs

- Complete Multipart Upload
- Upload Part (when the ILM rule uses the Strict or Balanced ingest behaviors)
- Upload Part - Copy (when the ILM rule uses the Strict or Balanced ingest behaviors)
- DELETE Object
- GET Object
- HEAD Object
- POST Object restore
- PUT Object
- PUT Object - Copy

Related information

[Operations on buckets](#)

[Operations on objects](#)

Benefits of active, idle, and concurrent HTTP connections

How you configure HTTP connections can impact the performance of the StorageGRID system. Configurations differ depending on whether the HTTP connection is active or idle or you have concurrent multiple connections.

You can identify the performance benefits for the following types of HTTP connections:

- Idle HTTP connections
- Active HTTP connections
- Concurrent HTTP connections

Related information

- [Benefits of keeping idle HTTP connections open](#)
- [Benefits of active HTTP connections](#)
- [Benefits of concurrent HTTP connections](#)
- [Separation of HTTP connection pools for read and write operations](#)

Benefits of keeping idle HTTP connections open

You should keep HTTP connections open even when client applications are idle to allow client applications to perform subsequent transactions over the open connection. Based on system measurements and integration experience, you should keep an idle HTTP connection open for a maximum of 10 minutes. StorageGRID might automatically close an HTTP connection that is kept open and idle for longer than 10 minutes.

Open and idle HTTP connections provide the following benefits:

- Reduced latency from the time that the StorageGRID system determines it has to perform an HTTP transaction to the time that the StorageGRID system can perform the transaction

Reduced latency is the main advantage, especially for the amount of time required to establish TCP/IP and TLS connections.

- Increased data transfer rate by priming the TCP/IP slow-start algorithm with previously performed transfers
- Instantaneous notification of several classes of fault conditions that interrupt connectivity between the client application and the StorageGRID system

Determining how long to keep an idle connection open is a trade-off between the benefits of slow start that is associated with the existing connection and the ideal allocation of the connection to internal system resources.

Benefits of active HTTP connections

For connections directly to Storage Nodes or to the CLB service (deprecated) on Gateway Nodes, you should limit the duration of an active HTTP connection to a maximum of 10 minutes, even if the HTTP connection continuously performs transactions.

Determining the maximum duration that a connection should be held open is a trade-off between the benefits of connection persistence and the ideal allocation of the connection to internal system resources.

For client connections to Storage Nodes or to the CLB service, limiting active HTTP connections provides the following benefits:

- Enables optimal load balancing across the StorageGRID system.

When using the CLB service, you should prevent long-lived TCP/IP connections to optimize load balancing across the StorageGRID system. You should configure client applications to track the duration of each HTTP connection and close the HTTP connection after a set time so that the HTTP connection can be reestablished and rebalanced.

The CLB service balances load across the StorageGRID system at the time that a client application establishes an HTTP connection. Over time, an HTTP connection might no longer be optimal as load balancing requirements change. The system performs its best load balancing when client applications

establish a separate HTTP connection for each transaction, but this negates the much more valuable gains associated with persistent connections.



The CLB service is deprecated.

- Allows client applications to direct HTTP transactions to LDR services that have available space.
- Allows maintenance procedures to start.

Some maintenance procedures start only after all the in-progress HTTP connections are complete.

For client connections to the Load Balancer service, limiting the duration of open connections can be useful for allowing some maintenance procedures to start promptly. If the duration of client connections is not limited, it may take several minutes for active connections to be automatically terminated.

Benefits of concurrent HTTP connections

You should keep multiple TCP/IP connections to the StorageGRID system open to allow parallelism, which increases performance. The optimal number of parallel connections depends on a variety of factors.

Concurrent HTTP connections provide the following benefits:

- Reduced latency

Transactions can start immediately instead of waiting for other transactions to be completed.

- Increased throughput

The StorageGRID system can perform parallel transactions and increase aggregate transaction throughput.

Client applications should establish multiple HTTP connections. When a client application has to perform a transaction, it can select and immediately use any established connection that is not currently processing a transaction.

Each StorageGRID system's topology has different peak throughput for concurrent transactions and connections before performance begins to degrade. Peak throughput depends on factors such as computing resources, network resources, storage resources, and WAN links. The number of servers and services and the number of applications that the StorageGRID system supports are also factors.

StorageGRID systems often support multiple client applications. You should keep this in mind when you determine the maximum number of concurrent connections used by a client application. If the client application consists of multiple software entities that each establish connections to the StorageGRID system, you should add up all the connections across the entities. You might have to adjust the maximum number of concurrent connections in the following situations:

- The StorageGRID system's topology affects the maximum number of concurrent transactions and connections that the system can support.
- Client applications that interact with the StorageGRID system over a network with limited bandwidth might have to reduce the degree of concurrency to ensure that individual transactions are completed in a reasonable time.
- When many client applications share the StorageGRID system, you might have to reduce the degree of

concurrency to avoid exceeding the limits of the system.

Separation of HTTP connection pools for read and write operations

You can use separate pools of HTTP connections for read and write operations and control how much of a pool to use for each. Separate pools of HTTP connections enable you to better control transactions and balance loads.

Client applications can create loads that are retrieve-dominant (read) or store-dominant (write). With separate pools of HTTP connections for read and write transactions, you can adjust how much of each pool to dedicate for read or write transactions.

Use Swift

Learn how client applications can use the OpenStack Swift API to interface with the StorageGRID system.

- [OpenStack Swift API support in StorageGRID](#)
- [Configuring tenant accounts and connections](#)
- [Swift REST API supported operations](#)
- [StorageGRID Swift REST API operations](#)
- [Configuring security for the REST API](#)
- [Monitoring and auditing operations](#)

OpenStack Swift API support in StorageGRID

StorageGRID supports the following specific versions of Swift and HTTP.

Item	Version
Swift specification	OpenStack Swift Object Storage API v1 as of November 2015
HTTP	1.1 For more information about HTTP, see HTTP/1.1 (RFCs 7230-35). Note: StorageGRID does not support HTTP/1.1 pipelining.

Related information

[OpenStack: Object Storage API](#)

History of Swift API support in StorageGRID

You should be aware of changes to the StorageGRID system's support for the Swift REST API.

Release	Comments
11.5	Removed Weak consistency control. The Available consistency level will be used instead.
11.4	Added support for TLS 1.3 and updated list of supported TLS cipher suites. CLB is deprecated. Added description of interrelationship between ILM and consistency setting.
11.3	Updated PUT Object operations to describe the impact of ILM rules that use synchronous placement at ingest (the Balanced and Strict options for Ingest Behavior). Added description of client connections that use load balancer endpoints or high availability groups. Updated list of supported TLS cipher suites. TLS 1.1 ciphers are no longer supported.
11.2	Minor editorial changes to document.
11.1	Added support for using HTTP for Swift client connections to grid nodes. Updated the definitions of consistency controls.
11.0	Added support for 1,000 containers for each tenant account.
10.3	Administrative updates and corrections to the document. Removed sections for configuring custom server certificates.
10.2	Initial support of the Swift API by the StorageGRID system. The currently supported version is OpenStack Swift Object Storage API v1.

How StorageGRID implements the Swift REST API

A client application can use Swift REST API calls to connect to Storage Nodes and Gateway Nodes to create containers and to store and retrieve objects. This enables service-oriented applications developed for OpenStack Swift to connect with on-premise object storage provided by the StorageGRID system.

Swift object management

After Swift objects have been ingested in the StorageGRID system, they are managed by the information lifecycle management (ILM) rules in the system's active ILM policy. The ILM rules and policy determine how StorageGRID creates and distributes copies of object data and how it manages those copies over time. For example, an ILM rule might apply to objects in specific Swift containers and might specify that multiple object copies be saved to several data centers for a certain number of years.

Contact your StorageGRID administrator if you need to understand how the grid's ILM rules and policies will affect the objects in your Swift tenant account.

Conflicting client requests

Conflicting client requests, such as a two clients writing to the same key, are resolved on a “latest-wins” basis. The timing for the “latest-wins” evaluation is based on when the StorageGRID system completes a given request, and not on when Swift clients begin an operation.

Consistency guarantees and controls

By default, StorageGRID provides read-after-write consistency for newly created objects and eventual consistency for object updates and HEAD operations. Any GET following a successfully completed PUT will be able to read the newly written data. Overwrites of existing objects, metadata updates, and deletes are eventually consistent. Overwrites generally take seconds or minutes to propagate, but can take up to 15 days.

StorageGRID also allows you to control consistency on a per container basis. You can change the consistency control to make a trade-off between the availability of the objects and the consistency of those objects across different Storage Nodes and sites, as required by your application.

Related information

[Manage objects with ILM](#)

[GET container consistency request](#)

[PUT container consistency request](#)

Recommendations for implementing the Swift REST API

You should follow these recommendations when implementing the Swift REST API for use with StorageGRID.

Recommendations for HEADs to non-existent objects

If your application routinely checks to see if an object exists at a path where you do not expect the object to actually exist, you should use the “Available” consistency control. For example, you should use the “Available” consistency control if your application performs a HEAD operation to a location before performing a PUT operation to that location.

Otherwise, if the HEAD operation does not find the object, you might receive a high number of 500 Internal Server errors if one or more Storage Nodes are unavailable.

You can set the “Available” consistency control for each container using the PUT container consistency request.

Recommendations for object names

You should not use random values as the first four characters of object names. Instead, you should use non-random, non-unique prefixes, such as image.

If you do need to use random and unique characters in object name prefixes, you should prefix the object names with a directory name. That is, use this format:

```
mycontainer/mydir/f8e3-image3132.jpg
```

Instead of this format:

```
mycontainer/f8e3-image3132.jpg
```

Recommendations for “range reads”

If the **Compress Stored Objects** option is selected (**Configuration > System Settings > Grid Options**), Swift client applications should avoid performing GET object operations that specify a range of bytes be returned. These “range read” operations are inefficient because StorageGRID must effectively uncompress the objects to access the requested bytes. GET Object operations that request a small range of bytes from a very large object are especially inefficient; for example, it is very inefficient to read a 10 MB range from a 50 GB compressed object.

If ranges are read from compressed objects, client requests can time out.



If you need to compress objects and your client application must use range reads, increase the read timeout for the application.

Related information

[GET container consistency request](#)

[PUT container consistency request](#)

[Administer StorageGRID](#)

Configuring tenant accounts and connections

Configuring StorageGRID to accept connections from client applications requires creating one or more tenant accounts and setting up the connections.

Creating and configuring Swift tenant accounts

A Swift tenant account is required before Swift API clients can store and retrieve objects on StorageGRID. Each tenant account has its own account ID, groups and users, and containers and objects.

Swift tenant accounts are created by a StorageGRID grid administrator using the Grid Manager or the Grid Management API.

When creating a Swift tenant account, the grid administrator specifies the following information:

- Display name for the tenant (the tenant’s account ID is assigned automatically and cannot be changed)
- Optionally, a storage quota for the tenant account—the maximum number of gigabytes, terabytes, or petabytes available for the tenant’s objects. A tenant’s storage quota represents a logical amount (object size), not a physical amount (size on disk).
- If single sign-on (SSO) is not in use for the StorageGRID system, whether the tenant account will use its own identity source or share the grid’s identity source, and the initial password for the tenant’s local root user.

- If SSO is enabled, which federated group has Root Access permission to configure the tenant account.

After a Swift tenant account is created, users with the Root Access permission can access the Tenant Manager to perform tasks such as the following:

- Setting up identity federation (unless the identity source is shared with the grid), and creating local groups and users
- Monitoring storage usage



Swift users must have the Root Access permission to access the Tenant Manager. However, the Root Access permission does not allow users to authenticate into the Swift REST API to create containers and ingest objects. Users must have the Swift Administrator permission to authenticate into the Swift REST API.

Related information

[Administer StorageGRID](#)

[Use a tenant account](#)

[Supported Swift API endpoints](#)

How client connections can be configured

A grid administrator makes configuration choices that affect how Swift clients connect to StorageGRID to store and retrieve data. The specific information you need to make a connection depends upon the configuration that was chosen.

Client applications can store or retrieve objects by connecting to any of the following:

- The Load Balancer service on Admin Nodes or Gateway Nodes, or optionally, the virtual IP address of a high availability (HA) group of Admin Nodes or Gateway Nodes
- The CLB service on Gateway Nodes, or optionally, the virtual IP address of a high availability group of Gateway Nodes



The CLB service is deprecated. Clients configured before the StorageGRID 11.3 release can continue to use the CLB service on Gateway Nodes. All other client applications that depend on StorageGRID to provide load balancing should connect using the Load Balancer service.

- Storage Nodes, with or without an external load balancer

When configuring StorageGRID, a grid administrator can use the Grid Manager or the Grid Management API to perform the following steps, all of which are optional:

1. Configure endpoints for the Load Balancer service.

You must configure endpoints to use the Load Balancer service. The Load Balancer service on Admin Nodes or Gateway Nodes distributes incoming network connections from client applications to Storage Nodes. When creating a load balancer endpoint, the StorageGRID administrator specifies a port number, whether the endpoint accepts HTTP or HTTPS connections, the type of client (S3 or Swift) that will use the endpoint, and the certificate to be used for HTTPS connections (if applicable).

2. Configure Untrusted Client Networks.

If a StorageGRID administrator configures a node's Client Network to be untrusted, the node only accepts inbound connections on the Client Network on ports that are explicitly configured as load balancer endpoints.

3. Configure high availability groups.

If an administrator creates an HA group, the network interfaces of multiple Admin Nodes or Gateway Nodes are placed into an active-backup configuration. Client connections are made using the virtual IP address of the HA group.

For more information about each option, see the instructions for administering StorageGRID.

Summary: IP addresses and ports for client connections

Client applications connect to StorageGRID using the IP address of a grid node and the port number of a service on that node. If high availability (HA) groups are configured, client applications can connect using the virtual IP address of the HA group.

Information required to make client connections

The table summarizes the different ways that clients can connect to StorageGRID and the IP addresses and ports that are used for each type of connection. Contact your StorageGRID administrator for more information, or see the instructions for administering StorageGRID for a description of how to find this information in the Grid Manager.

Where connection is made	Service that client connects to	IP address	Port
HA group	Load Balancer	Virtual IP address of an HA group	<ul style="list-style-type: none">• Load balancer endpoint port
HA group	CLB Note: The CLB service is deprecated.	Virtual IP address of an HA group	Default Swift ports: <ul style="list-style-type: none">• HTTPS: 8083• HTTP: 8085
Admin Node	Load Balancer	IP address of the Admin Node	<ul style="list-style-type: none">• Load balancer endpoint port
Gateway Node	Load Balancer	IP address of the Gateway Node	<ul style="list-style-type: none">• Load balancer endpoint port
Gateway Node	CLB Note: The CLB service is deprecated.	IP address of the Gateway Node Note: By default, HTTP ports for CLB and LDR are not enabled.	Default Swift ports: <ul style="list-style-type: none">• HTTPS: 8083• HTTP: 8085

Where connection is made	Service that client connects to	IP address	Port
Storage Node	LDR	IP address of Storage Node	Default Swift ports: <ul style="list-style-type: none"> • HTTPS: 18083 • HTTP: 18085

Example

To connect a Swift client to the Load Balancer endpoint of an HA group of Gateway Nodes, use a URL structured as shown below:

- `https://VIP-of-HA-group:LB-endpoint-port`

For example, if the virtual IP address of the HA group is 192.0.2.6 and the port number of a Swift Load Balancer endpoint is 10444, then a Swift client could use the following URL to connect to StorageGRID:

- `https://192.0.2.6:10444`

It is possible to configure a DNS name for the IP address that clients use to connect to StorageGRID. Contact your local network administrator.

Deciding to use HTTPS or HTTP connections

When client connections are made using a Load Balancer endpoint, connections must be made using the protocol (HTTP or HTTPS) that was specified for that endpoint. To use HTTP for client connections to Storage Nodes or to the CLB service on Gateway Nodes, you must enable its use.

By default, when client applications connect to Storage Nodes or the CLB service on Gateway Nodes, they must use encrypted HTTPS for all connections. Optionally, you can enable less-secure HTTP connections by selecting the **Enable HTTP Connection** grid option in the Grid Manager. For example, a client application might use HTTP when testing the connection to a Storage Node in a non-production environment.



Be careful when enabling HTTP for a production grid since requests will be sent unencrypted.



The CLB service is deprecated.

If the **Enable HTTP Connection** option is selected, clients must use different ports for HTTP than they use for HTTPS. See the instructions for administering StorageGRID.

Related information

[Administer StorageGRID](#)

Testing your connection in the Swift API configuration

You can use the Swift CLI to test your connection to the StorageGRID system and to verify that you can read and write objects to the system.

What you'll need

- You must have downloaded and installed `python-swiftclient`, the Swift command-line client.

- You must have a Swift tenant account in the StorageGRID system.

About this task

If you have not configured security, you must add the `--insecure` flag to each of these commands.

Steps

1. Query the info URL for your StorageGRID Swift deployment:

```
swift
-U <Tenant_Account_ID:Account_User_Name>
-K <User_Password>
-A https://<FQDN | IP>:<Port>/info
capabilities
```

This is sufficient to test that your Swift deployment is functional. To further test account configuration by storing an object, continue with the additional steps.

2. Put an object in the container:

```
touch test_object
swift
-U <Tenant_Account_ID:Account_User_Name>
-K <User_Password>
-A https://<FQDN | IP>:<Port>/auth/v1.0
upload test_container test_object
--object-name test_object
```

3. Get the container to verify the object:

```
swift
-U <Tenant_Account_ID:Account_User_Name>
-K <User_Password>
-A https://<FQDN | IP>:<Port>/auth/v1.0
list test_container
```

4. Delete the object:

```
swift
-U <Tenant_Account_ID:Account_User_Name>
-K <User_Password>
-A https://<FQDN | IP>:<Port>/auth/v1.0
delete test_container test_object
```


5. Delete the container:

```
swift
-U `<_Tenant_Account_ID:Account_User_Name_>`
-K `<_User_Password_>`
-A `https://<_FQDN_ | _IP_>:<_Port_>/auth/v1.0'
delete test_container
```

Related information

[Creating and configuring Swift tenant accounts](#)

[Configuring security for the REST API](#)

Swift REST API supported operations

The StorageGRID system supports most operations in the OpenStack Swift API. Before integrating Swift REST API clients with StorageGRID, review the implementation details for account, container, and object operations.

Operations supported in StorageGRID

The following Swift API operations are supported:

- [Account operations](#)
- [Container operations](#)
- [Object operations](#)

Common response headers for all operations

The StorageGRID system implements all common headers for supported operations as defined by the OpenStack Swift Object Storage API v1.

Related information

[OpenStack: Object Storage API](#)

Supported Swift API endpoints

StorageGRID supports the following Swift API endpoints: the info URL, the auth URL, and the storage URL.

info URL

You can determine the capabilities and limitations of the StorageGRID Swift implementation by issuing a GET request to the Swift base URL with the /info path.

```
https://FQDN | Node IP:Swift Port/info/
```

In the request:

- *FQDN* is the fully qualified domain name.
- *Node IP* is the IP address for the Storage Node or the Gateway Node on the StorageGRID network.
- *Swift Port* is the port number used for Swift API connections on the Storage Node or Gateway Node.

For example, the following info URL would request information from a Storage Node with the IP address of 10.99.106.103 and using port 18083.

```
https://10.99.106.103:18083/info/
```

The response includes the capabilities of the Swift implementation as a JSON dictionary. A client tool can parse the JSON response to determine the capabilities of the implementation and use them as constraints for subsequent storage operations.

The StorageGRID implementation of Swift allows unauthenticated access to the info URL.

auth URL

A client can use the Swift auth URL to authenticate as a tenant account user.

```
https://FQDN | Node IP:Swift Port/auth/v1.0/
```

You must provide the tenant account ID, user name, and password as parameters in the X-Auth-User and X-Auth-Key request headers, as follows:

```
X-Auth-User: Tenant_Account_ID:Username
```

```
X-Auth-Key: Password
```

In the request headers:

- *Tenant_Account_ID* is the account ID assigned by StorageGRID when the Swift tenant was created. This is the same tenant account ID used on the Tenant Manager sign-in page.
- *Username* is the name of a tenant user that has been created in the Tenant Manager. This user must belong to a group that has the Swift Administrator permission. The tenant's root user cannot be configured to use the Swift REST API.

If Identity Federation is enabled for the tenant account, provide the username and password of the federated user from the LDAP server. Alternatively, provide the LDAP user's domain name. For example:

```
X-Auth-User: Tenant_Account_ID:Username@Domain_Name
```

- *Password* is the password for the tenant user. User passwords are created and managed in the Tenant Manager.

The response to a successful authentication request returns a storage URL and an auth token, as follows:

```
X-Storage-Url: https://FQDN | Node_IP:Swift_Port/v1/Tenant_Account_ID
```

```
X-Auth-Token: token
```

```
X-Storage-Token: token
```

By default, the token is valid for 24 hours from generation time.

Tokens are generated for a specific tenant account. A valid token for one account does not authorize a user to access another account.

storage URL

A client application can issue Swift REST API calls to perform supported account, container, and object operations against a Gateway Node or Storage Node. Storage requests are addressed to the storage URL returned in the authentication response. The request must also include the X-Auth-Token header and value returned from the auth request.

```
https://FQDN | IP:Swift_Port/v1/Tenant_Account_ID
```

```
[/container] [/object]
```

```
X-Auth-Token: token
```

Some storage response headers that contain usage statistics might not reflect accurate numbers for recently modified objects. It might take a few minutes for accurate numbers to appear in these headers.

The following response headers for account and container operations are examples of those that contain usage statistics:

- X-Account-Bytes-Used
- X-Account-Object-Count
- X-Container-Bytes-Used
- X-Container-Object-Count

Related information

[How client connections can be configured](#)

[Creating and configuring Swift tenant accounts](#)

[Account operations](#)

[Container operations](#)

[Object operations](#)

Account operations

The following Swift API operations are performed on accounts.

GET account

This operation retrieves the container list associated with the account and account usage statistics.

The following request parameter is required:

- Account

The following request header is required:

- X-Auth-Token

The following supported request query parameters are optional:

- Delimiter
- End_marker
- Format
- Limit
- Marker
- Prefix

A successful execution returns the following headers with an “HTTP/1.1 204 No Content” response if the account is found and has no containers or the container list is empty; or an “HTTP/1.1 200 OK” response if the account is found and the container list is not empty:

- Accept-Ranges
- Content-Length
- Content-Type
- Date
- X-Account-Bytes-Used
- X-Account-Container-Count
- X-Account-Object-Count
- X-Timestamp
- X-Trans-Id

HEAD account

This operation retrieves account information and statistics from a Swift account.

The following request parameter is required:

- Account

The following request header is required:

- X-Auth-Token

A successful execution returns the following headers with an “HTTP/1.1 204 No Content” response:

- Accept-Ranges
- Content-Length
- Date
- X-Account-Bytes-Used
- X-Account-Container-Count

- X-Account-Object-Count
- X-Timestamp
- X-Trans-Id

Related information

[Swift operations tracked in the audit logs](#)

Container operations

StorageGRID supports a maximum of 1,000 containers per Swift account. The following Swift API operations are performed on containers.

DELETE container

This operation removes an empty container from a Swift account in a StorageGRID system.

The following request parameters are required:

- Account
- Container

The following request header is required:

- X-Auth-Token

A successful execution returns the following headers with an "HTTP/1.1 204 No Content" response:

- Content-Length
- Content-Type
- Date
- X-Trans-Id

GET container

This operation retrieves the object list associated with the container along with container statistics and metadata in a StorageGRID system.

The following request parameters are required:

- Account
- Container

The following request header is required:

- X-Auth-Token

The following supported request query parameters are optional:

- Delimiter

- End_marker
- Format
- Limit
- Marker
- Path
- Prefix

A successful execution returns the following headers with an "HTTP/1.1 200 Success" or a "HTTP/1.1 204 No Content" response:

- Accept-Ranges
- Content-Length
- Content-Type
- Date
- X-Container-Bytes-Used
- X-Container-Object-Count
- X-Timestamp
- X-Trans-Id

HEAD container

This operation retrieves container statistics and metadata from a StorageGRID system.

The following request parameters are required:

- Account
- Container

The following request header is required:

- X-Auth-Token

A successful execution returns the following headers with an "HTTP/1.1 204 No Content" response:

- Accept-Ranges
- Content-Length
- Date
- X-Container-Bytes-Used
- X-Container-Object-Count
- X-Timestamp
- X-Trans-Id

PUT container

This operation creates a container for an account in a StorageGRID system.

The following request parameters are required:

- Account
- Container

The following request header is required:

- X-Auth-Token

A successful execution returns the following headers with an "HTTP/1.1 201 Created" or "HTTP/1.1 202 Accepted" (if the container already exists under this account) response:

- Content-Length
- Date
- X-Timestamp
- X-Trans-Id

A container name must be unique in the StorageGRID namespace. If the container exists under another account, the following header is returned: "HTTP/1.1 409 Conflict."

Related information

[Swift operations tracked in the audit logs](#)

Object operations

The following Swift API operations are performed on objects.

DELETE object

This operation deletes an object's content and metadata from the StorageGRID system.

The following request parameters are required:

- Account
- Container
- Object

The following request header is required:

- X-Auth-Token

A successful execution returns the following response headers with an HTTP/1.1 204 No Content response:

- Content-Length
- Content-Type

- Date
- X-Trans-Id

When processing a DELETE Object request, StorageGRID attempts to immediately remove all copies of the object from all stored locations. If successful, StorageGRID returns a response to the client immediately. If all copies cannot be removed within 30 seconds (for example, because a location is temporarily unavailable), StorageGRID queues the copies for removal and then indicates success to the client.

For more information on how objects are deleted, see the instructions for managing objects with information lifecycle management.

GET object

This operation retrieves the object content and gets the object metadata from a StorageGRID system.

The following request parameters are required:

- Account
- Container
- Object

The following request header is required:

- X-Auth-Token

The following request headers are optional:

- Accept-Encoding
- If-Match
- If-Modified-Since
- If-None-Match
- If-Unmodified-Since
- Range

A successful execution returns the following headers with an HTTP/1.1 200 OK response:

- Accept-Ranges
- Content-Disposition, returned only if Content-Disposition metadata was set
- Content-Encoding, returned only if Content-Encoding metadata was set
- Content-Length
- Content-Type
- Date
- ETag
- Last-Modified

- X-Timestamp
- X-Trans-Id

HEAD object

This operation retrieves metadata and properties of an ingested object from a StorageGRID system.

The following request parameters are required:

- Account
- Container
- Object

The following request header is required:

- X-Auth-Token

A successful execution returns the following headers with an "HTTP/1.1 200 OK" response:

- Accept-Ranges
- Content-Disposition, **returned only if Content-Disposition metadata was set**
- Content-Encoding, **returned only if Content-Encoding metadata was set**
- Content-Length
- Content-Type
- Date
- ETag
- Last-Modified
- X-Timestamp
- X-Trans-Id

PUT object

This operation creates a new object with data and metadata, or replaces an existing object with data and metadata in a StorageGRID system.

StorageGRID supports objects up to 5 TB in size.



Conflicting client requests, such as a two clients writing to the same key, are resolved on a “latest-wins” basis. The timing for the “latest-wins” evaluation is based on when the StorageGRID system completes a given request, and not on when Swift clients begin an operation.

The following request parameters are required:

- Account
- Container

- Object

The following request header is required:

- X-Auth-Token

The following request headers are optional:

- Content-Disposition
- Content-Encoding

Do not use chunked Content-Encoding if the ILM rule that applies to an object filters objects based on size and uses synchronous placement on ingest (the Balanced or Strict options for Ingest Behavior).

- Transfer-Encoding

Do not use compressed or chunked Transfer-Encoding if the ILM rule that applies to an object filters objects based on size and uses synchronous placement on ingest (the Balanced or Strict options for Ingest Behavior).

- Content-Length

If an ILM rule filters objects by size and uses synchronous placement on ingest, you must specify Content-Length.



If you do not follow these guidelines for Content-Encoding, Transfer-Encoding, and Content-Length, StorageGRID must save the object before it can determine object size and apply the ILM rule. In other words, StorageGRID must default to creating interim copies of an object on ingest. That is, StorageGRID must use the Dual Commit option for Ingest Behavior.

For more information about synchronous placement and ILM rules, see the instructions for managing objects with information lifecycle management.

- Content-Type
- ETag
- X-Object-Meta-`<name\>` (object-related metadata)

If you want to use the **User Defined Creation Time** option as the Reference Time for an ILM rule, you must store the value in a user-defined header named X-Object-Meta-Creation-Time. For example:

```
X-Object-Meta-Creation-Time: 1443399726
```

This field is evaluated as seconds since January 1, 1970.

- X-Storage-Class: `reduced_redundancy`

This header affects how many object copies StorageGRID creates if the ILM rule that matches an ingested object specifies an Ingest Behavior of Dual Commit or Balanced.

- **Dual commit:** If the ILM rule specifies the Dual commit option for Ingest Behavior, StorageGRID creates a single interim copy as the object is ingested (single commit).
- **Balanced:** If the ILM rule specifies the Balanced option, StorageGRID makes a single interim copy only if the system cannot immediately make all copies specified in the rule. If StorageGRID can perform synchronous placement, this header has no effect.

The `reduced_redundancy` header is best used when the ILM rule that matches the object creates a single replicated copy. In this case using `reduced_redundancy` eliminates the unnecessary creation and deletion of an extra object copy for every ingest operation.

Using the `reduced_redundancy` header is not recommended in other circumstances because it increases the risk the loss of object data during ingest. For example, you might lose data if the single copy is initially stored on a Storage Node that fails before ILM evaluation can occur.



Having only one replicated copy for any time period puts data at risk of permanent loss. If only one replicated copy of an object exists, that object is lost if a Storage Node fails or has a significant error. You also temporarily lose access to the object during maintenance procedures such as upgrades.

Note that specifying `reduced_redundancy` only affects how many copies are created when an object is first ingested. It does not affect how many copies of the object are made when the object is evaluated by the active ILM policy and does not result in data being stored at lower levels of redundancy in the StorageGRID system.

A successful execution returns the following headers with an "HTTP/1.1 201 Created" response:

- Content-Length
- Content-Type
- Date
- ETag
- Last-Modified
- X-Trans-Id

Related information

[Manage objects with ILM](#)

[Swift operations tracked in the audit logs](#)

OPTIONS request

The OPTIONS request checks the availability of an individual Swift service. The OPTIONS request is processed by the Storage Node or Gateway Node specified in the URL.

OPTIONS method

For example, client applications can issue an OPTIONS request to the Swift port on a Storage Node, without providing Swift authentication credentials, to determine whether the Storage Node is available. You can use this request for monitoring or to allow external load balancers to identify when a Storage Node is down.

When used with the info URL or the storage URL, the OPTIONS method returns a list of supported verbs for the given URL (for example, HEAD, GET, OPTIONS, and PUT). The OPTIONS method cannot be used with the auth URL.

The following request parameter is required:

- Account

The following request parameters are optional:

- Container
- Object

A successful execution returns the following headers with an “HTTP/1.1 204 No Content” response. The OPTIONS request to the storage URL does not require that the target exists.

- Allow (a list of supported verbs for the given URL, for example, HEAD, GET, OPTIONS, and PUT)
- Content-Length
- Content-Type
- Date
- X-Trans-Id

Related information

[Supported Swift API endpoints](#)

Error responses to Swift API operations

Understanding the possible error responses can help you troubleshoot operations.

The following HTTP status codes might be returned when errors occur during an operation:

Swift error name	HTTP status
AccountNameTooLong, ContainerNameTooLong, HeaderTooBig, InvalidContainerName, InvalidRequest, InvalidURI, MetadataNameTooLong, MetadataValueTooBig, MissingSecurityHeader, ObjectNameTooLong, TooManyContainers, TooManyMetadataItems, TotalMetadataTooLarge	400 Bad Request
AccessDenied	403 Forbidden
ContainerNotEmpty, ContainerAlreadyExists	409 Conflict
InternalError	500 Internal Server Error
InvalidRange	416 Requested Range Not Satisfiable

Swift error name	HTTP status
MethodNotAllowed	405 Method Not Allowed
MissingContentLength	411 Length Required
NotFound	404 Not Found
NotImplemented	501 Not Implemented
PreconditionFailed	412 Precondition Failed
ResourceNotFound	404 Not Found
Unauthorized	401 Unauthorized
UnprocessableEntity	422 Unprocessable Entity

StorageGRID Swift REST API operations

There are operations added on to the Swift REST API that are specific to StorageGRID system.

GET container consistency request

Consistency level makes a trade-off between the availability of the objects and the consistency of those objects across different Storage Nodes and sites. The GET container consistency request allows you to determine the consistency level being applied to a particular container.

Request

Request HTTP Header	Description
X-Auth-Token	Specifies the Swift authentication token for the account to use for the request.
x-ntap-sg-consistency	Specifies the type of request, where <code>true</code> = GET container consistency, and <code>false</code> = GET container.
Host	The hostname to which the request is directed.

Request example

```
GET /v1/28544923908243208806/Swift container
X-Auth-Token: SGRD_3a877009a2d24cb1801587bfa9050f29
x-ntap-sg-consistency: true
Host: test.com
```

Response

Response HTTP Header	Description
Date	The date and time of the response.
Connection	Whether the connection to the server is open or closed.
X-Trans-Id	The unique transaction identifier for the request.
Content-Length	The length of the response body.
x-ntap-sg-consistency	<p>The consistency control level being applied to the container. The following values are supported:</p> <ul style="list-style-type: none">• all: All nodes receive the data immediately or the request will fail.• strong-global: Guarantees read-after-write consistency for all client requests across all sites.• strong-site: Guarantees read-after-write consistency for all client requests within a site.• read-after-new-write: Provides read-after-write consistency for new objects and eventual consistency for object updates. Offers high availability and data protection guarantees. <p>Note: If your application uses HEAD requests on objects that do not exist, you might receive a high number of 500 Internal Server errors if one or more Storage Nodes are unavailable. To prevent these errors, use the “available” level.</p> <ul style="list-style-type: none">• available (eventual consistency for HEAD operations): Behaves the same as the “read-after-new-write” consistency level, but only provides eventual consistency for HEAD operations. Offers higher availability for HEAD operations than “read-after-new-write” if Storage Nodes are unavailable.

Response example

```
HTTP/1.1 204 No Content
Date: Sat, 29 Nov 2015 01:02:18 GMT
Connection: CLOSE
X-Trans-Id: 1936575373
Content-Length: 0
x-ntap-sg-consistency: strong-site
```

Related information

[Use a tenant account](#)

PUT container consistency request

The PUT container consistency request allows you to specify the consistency level to apply to operations performed on a container. By default, new containers are created using the “read-after-new-write” consistency level.

Request

Request HTTP Header	Description
X-Auth-Token	The Swift authentication token for the account to use for the request.

Request HTTP Header	Description
x-ntap-sg-consistency	<p>The consistency control level to apply to operations on the container. The following values are supported:</p> <ul style="list-style-type: none"> • all: All nodes receive the data immediately or the request will fail. • strong-global: Guarantees read-after-write consistency for all client requests across all sites. • strong-site: Guarantees read-after-write consistency for all client requests within a site. • read-after-new-write: Provides read-after-write consistency for new objects and eventual consistency for object updates. Offers high availability and data protection guarantees. <p>Note: If your application uses HEAD requests on objects that do not exist, you might receive a high number of 500 Internal Server errors if one or more Storage Nodes are unavailable. To prevent these errors, use the “available” level.</p> <ul style="list-style-type: none"> • available (eventual consistency for HEAD operations): Behaves the same as the “read-after-new-write” consistency level, but only provides eventual consistency for HEAD operations. Offers higher availability for HEAD operations than “read-after-new-write” if Storage Nodes are unavailable.
Host	The hostname to which the request is directed.

How consistency controls and ILM rules interact to affect data protection

Both your choice of consistency control and your ILM rule affect how objects are protected. These settings can interact.

For example, the consistency control used when an object is stored affects the initial placement of object metadata, while the ingest behavior selected for the ILM rule affects the initial placement of object copies. Because StorageGRID requires access to both an object’s metadata and its data to fulfill client requests, selecting matching levels of protection for the consistency level and ingest behavior can provide better initial data protection and more predictable system responses.

The following ingest behaviors are available for ILM rules:

- **Strict**: All copies specified in the ILM rule must be made before success is returned to the client.
- **Balanced**: StorageGRID attempts to make all copies specified in the ILM rule at ingest; if this is not possible, interim copies are made and success is returned to the client. The copies specified in the ILM rule are made when possible.
- **Dual Commit**: StorageGRID immediately makes interim copies of the object and returns success to the client. Copies specified in the ILM rule are made when possible.



Before selecting the ingest behavior for an ILM rule, read the full description of these settings in the instructions for managing objects with information lifecycle management.

Example of how the consistency control and ILM rule can interact

Suppose you have a two-site grid with the following ILM rule and the following consistency level setting:

- **ILM rule:** Create two object copies, one at the local site and one at a remote site. The Strict ingest behavior is selected.
- **Consistency level:** “strong-global” (Object metadata is immediately distributed to all sites.)

When a client stores an object to the grid, StorageGRID makes both object copies and distributes metadata to both sites before returning success to the client.

The object is fully protected against loss at the time of the ingest successful message. For example, if the local site is lost shortly after ingest, copies of both the object data and the object metadata still exist at the remote site. The object is fully retrievable.

If you instead used the same ILM rule and the “strong-site” consistency level, the client might receive a success message after object data is replicated to the remote site but before object metadata is distributed there. In this case, the level of protection of object metadata does not match the level of protection for object data. If the local site is lost shortly after ingest, object metadata is lost. The object cannot be retrieved.

The inter-relationship between consistency levels and ILM rules can be complex. Contact NetApp if you require assistance.

Request example

```
PUT /v1/28544923908243208806/_Swift container_  
X-Auth-Token: SGRD_3a877009a2d24cb1801587bfa9050f29  
x-ntap-sg-consistency: strong-site  
Host: test.com
```

Response

Response HTTP Header	Description
Date	The date and time of the response.
Connection	Whether the connection to the server is open or closed.
X-Trans-Id	The unique transaction identifier for the request.
Content-Length	The length of the response body.

Response example

```
HTTP/1.1 204 No Content
Date: Sat, 29 Nov 2015 01:02:18 GMT
Connection: CLOSE
X-Trans-Id: 1936575373
Content-Length: 0
```

Related information

[Use a tenant account](#)

Configuring security for the REST API

You should review the security measures implemented for the REST API and understand how to secure your system.

How StorageGRID provides security for the REST API

You should understand how the StorageGRID system implements security, authentication, and authorization for the REST API.

StorageGRID uses the following security measures.

- Client communications with the Load Balancer service use HTTPS if HTTPS is configured for the load balancer endpoint.

When you configure a load balancer endpoint, HTTP can optionally be enabled. For example, you might want to use HTTP for testing or other non-production purposes. See the instructions for administering StorageGRID for more information.

- By default, StorageGRID uses HTTPS for client communications with Storage Nodes and the CLB service on Gateway Nodes.

HTTP can optionally be enabled for these connections. For example, you might want to use HTTP for testing or other non-production purposes. See the instructions for administering StorageGRID for more information.



The CLB service is deprecated.

- Communications between StorageGRID and the client are encrypted using TLS.
- Communications between the Load Balancer service and Storage Nodes within the grid are encrypted whether the load balancer endpoint is configured to accept HTTP or HTTPS connections.
- Clients must supply HTTP authentication headers to StorageGRID to perform REST API operations.

Security certificates and client applications

Clients can connect to the Load Balancer service on Gateway Nodes or Admin Nodes, directly to Storage Nodes, or to the CLB service on Gateway Nodes.

In all cases, client applications can make TLS connections using either a custom server certificate uploaded by the grid administrator or a certificate generated by the StorageGRID system:

- When client applications connect to the Load Balancer service, they do so using the certificate that was configured for the specific load balancer endpoint used to make the connection. Each endpoint has its own certificate, which is either a custom server certificate uploaded by the grid administrator or a certificate that the grid administrator generated in StorageGRID when configuring the endpoint.
- When client applications connect directly to a Storage Node or to the CLB service on Gateway Nodes, they use either the system-generated server certificates that were generated for Storage Nodes when the StorageGRID system was installed (which are signed by the system certificate authority), or a single custom server certificate that is supplied for the grid by a grid administrator.

Clients should be configured to trust the certificate authority that signed whichever certificate they use to establish TLS connections.

See the instructions for administering StorageGRID for information on configuring load balancer endpoints, and for instructions on adding a single custom server certificate for TLS connections directly to Storage Nodes or to the CLB service on Gateway Nodes.

Summary

The following table shows how security issues are implemented in the S3 and Swift REST APIs:

Security issue	Implementation for REST API
Connection security	TLS
Server authentication	X.509 server certificate signed by system CA or custom server certificate supplied by administrator
Client authentication	<ul style="list-style-type: none"> • S3: S3 account (access key ID and secret access key) • Swift: Swift account (user name and password)
Client authorization	<ul style="list-style-type: none"> • S3: Bucket ownership and all applicable access control policies • Swift: Administrator role access

Related information

[Administer StorageGRID](#)

Supported hashing and encryption algorithms for TLS libraries

The StorageGRID system supports a limited set of cipher suites that client applications can use when establishing a Transport Layer Security (TLS) session.

Supported versions of TLS

StorageGRID supports TLS 1.2 and TLS 1.3.



SSLv3 and TLS 1.1 (or earlier versions) are no longer supported.

Supported cipher suites

TLS version	IANA name of cipher suite
1.2	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
	TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
1.3	TLS_AES_256_GCM_SHA384
	TLS_CHACHA20_POLY1305_SHA256
	TLS_AES_128_GCM_SHA256

Deprecated cipher suites

The following cipher suites are deprecated. Support for these ciphers will be removed in a future release.

IANA Name
TLS_RSA_WITH_AES_128_GCM_SHA256
TLS_RSA_WITH_AES_256_GCM_SHA384

Related information

[How client connections can be configured](#)

Monitoring and auditing operations

You can monitor workloads and efficiencies for client operations by viewing transaction trends for the entire grid, or for specific nodes. You can use audit messages to monitor client operations and transactions.

Monitoring object ingest and retrieval rates

You can monitor object ingest and retrieval rates as well as metrics for object counts, queries, and verification. You can view the number of successful and failed attempts by client applications to read, write, and modify objects in the StorageGRID system.

Steps

1. Sign in to the Grid Manager using a supported browser.
2. On the Dashboard, locate the Protocol Operations section.

This section summarizes the number of client operations performed by your StorageGRID system. Protocol rates are averaged over the last two minutes.

3. Select **Nodes**.

4. From the Nodes home page (deployment level), click the **Load Balancer** tab.

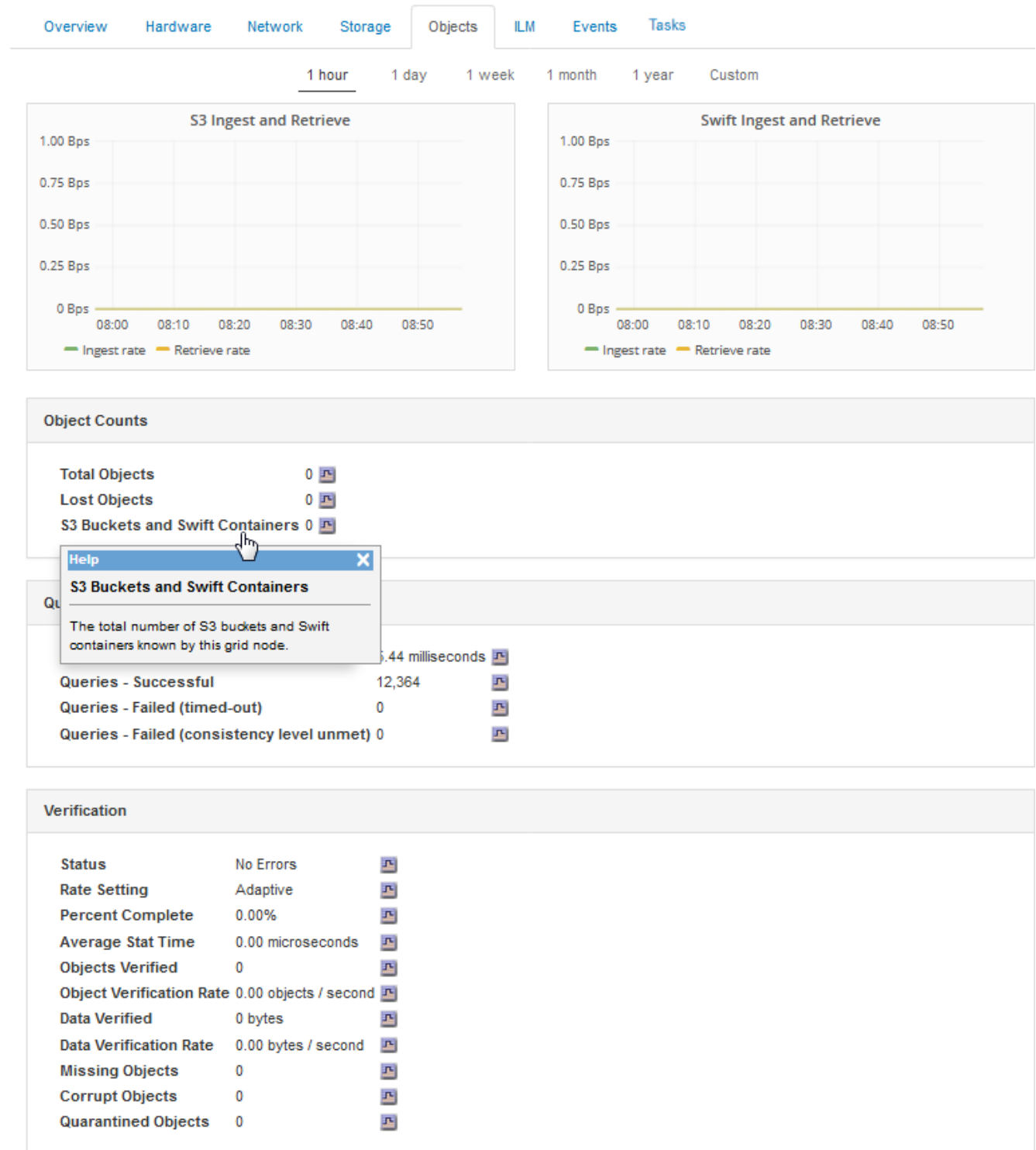
The charts show trends for all client traffic directed to load balancer endpoints within the grid. You can select a time interval in hours, days, weeks, months, or years, or you can apply a custom interval.

5. From the Nodes home page (deployment level), click the **Objects** tab.

The chart shows ingest and retrieve rates for your entire StorageGRID system in bytes per second and total bytes. You can select a time interval in hours, days, weeks, months, or years, or you can apply a custom interval.

6. To see information for a particular Storage Node, select the node from the list on the left, and click the **Objects** tab.

The chart shows the object ingest and retrieval rates for this Storage Node. The tab also includes metrics for object counts, queries, and verification. You can click the labels to see the definitions of these metrics.



7. If you want even more detail:

- Select **Support > Tools > Grid Topology**.
- Select **site > Overview > Main**.

The API Operations section displays summary information for the entire grid.

- Select **Storage Node > LDR > client application > Overview > Main**

The Operations section displays summary information for the selected Storage Node.

Accessing and reviewing audit logs

Audit messages are generated by StorageGRID services and stored in text log files. API-specific audit messages in the audit logs provide critical security, operation, and performance monitoring data that can help you evaluate the health of your system.

What you'll need

- You must have specific access permissions.
- You must have the `Passwords.txt` file.
- You must know the IP address of an Admin Node.

About this task

The active audit log file is named `audit.log`, and it is stored on Admin Nodes.

Once a day, the active `audit.log` file is saved, and a new `audit.log` file is started. The name of the saved file indicates when it was saved, in the format `yyyy-mm-dd.txt`.

After a day, the saved file is compressed and renamed, in the format `yyyy-mm-dd.txt.gz`, which preserves the original date.

This example shows the active `audit.log` file, the previous day's file (`2018-04-15.txt`), and the compressed file for the prior day (`2018-04-14.txt.gz`).

```
audit.log
2018-04-15.txt
2018-04-14.txt.gz
```

Steps

1. Log in to an Admin Node:
 - a. Enter the following command: `ssh admin@primary_Admin_Node_IP`
 - b. Enter the password listed in the `Passwords.txt` file.
2. Go to the directory containing the audit log files: `cd /var/local/audit/export`
3. View the current or a saved audit log file, as required.

Related information

[Review audit logs](#)

Swift operations tracked in the audit logs

All successful storage DELETE, GET, HEAD, POST, and PUT operations are tracked in the StorageGRID audit log. Failures are not logged, nor are info, auth, or OPTIONS requests.

See *Understanding audit messages* for details about the information tracked for the following Swift operations.

Account operations

- GET account
- HEAD account

Container operations

- DELETE container
- GET container
- HEAD container
- PUT container

Object operations

- DELETE object
- GET object
- HEAD object
- PUT object

Related information

[Review audit logs](#)

[Account operations](#)

[Container operations](#)

[Object operations](#)

Copyright information

Copyright © 2025 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

Trademark information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.