Use StorageGRID

StorageGRID software

NetApp
December 03, 2025

Table of Contents

Use StorageGRID tenants and clients

Use a tenant account

Use a 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.

What is a tenant account?

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

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 don't need to create separate tenant accounts. See instructions for implementing S3 buckets and bucket policies for more information.

- **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.

How to create a tenant account

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

- Basic information including the tenant name, client type (S3) and optional storage quota.
- Permissions for the tenant account, such as whether the tenant account can use S3 platform services, configure its own identity source, use S3 Select, or use a grid federation connection.
- The initial root access for the tenant, based on whether the StorageGRID system uses local groups and users, identity federation, or single sign-on (SSO).

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.

Configure S3 tenants

After an S3 tenant account is created, you 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)

- Manage groups and users
- Use grid federation for account clone and cross-grid replication
- Manage S3 access keys
- Create and manage S3 buckets
- Use S3 platform services
- Use S3 Select
- Monitor storage usage

Although you can create and manage S3 buckets with the Tenant Manager, you must use an S3 client or S3 Console to ingest and manage objects.

How to sign in and sign out

Sign in to Tenant Manager

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

Before you begin

- You have your login credentials.
- You 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 includes a fully qualified domain name (FQDN), the IP address of an Admin Node, or the virtual IP address of an HA group of Admin Nodes. It might 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 have this account ID.
- You are using a supported web browser.
- Cookies are enabled in your web browser.
- You belong to a user group that has 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.

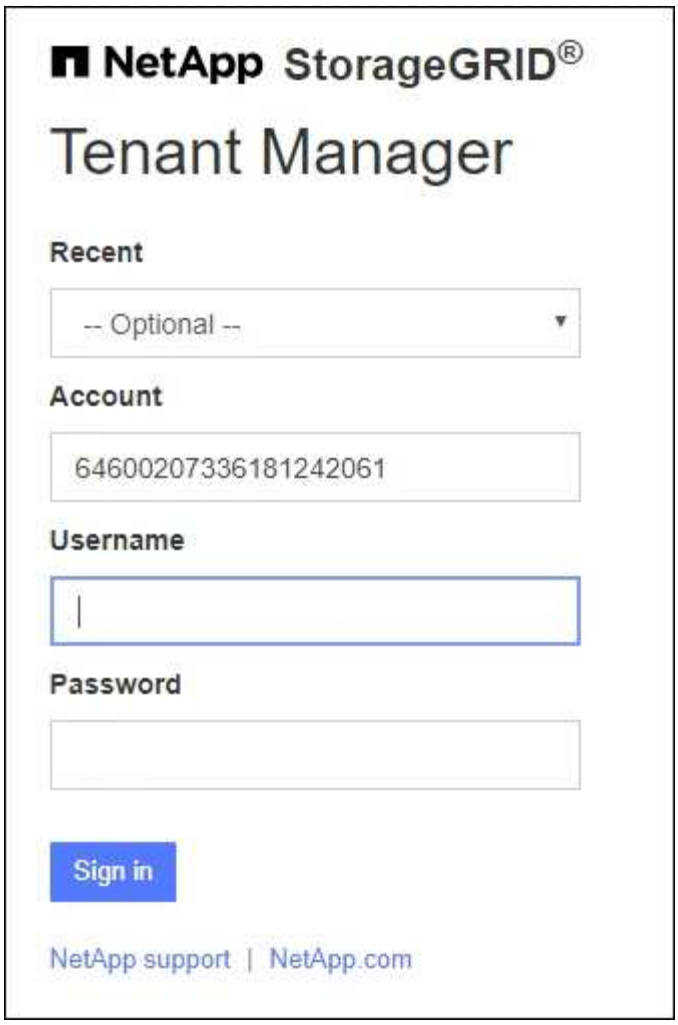
The sign-in screen that appears depends on the URL you entered and whether single sign-on (SSO) has been configured for StorageGRID.

Not using SSO

If StorageGRID is not using SSO, one of the following screens appears:

- The Grid Manager sign-in page. Select the **Tenant sign-in** link.

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



a. 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.
b. Enter your username and password.
c. Select **Sign in**.

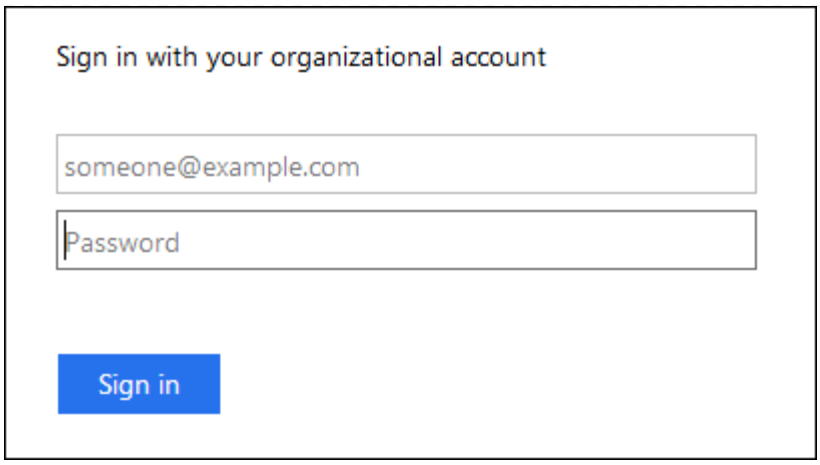
   The Tenant Manager dashboard appears.

d. If you received an initial password from someone else, select ***username*** > **Change password** to secure your account.

Using SSO

If StorageGRID is using SSO, one of the following screens appears:

- Your organization's SSO page. For example:



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

- The Tenant Manager SSO sign-in page.

a. 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.
b. Select **Sign in**.
c. Sign in with your standard SSO credentials on your organization's SSO sign-in page.

   The Tenant Manager dashboard appears.

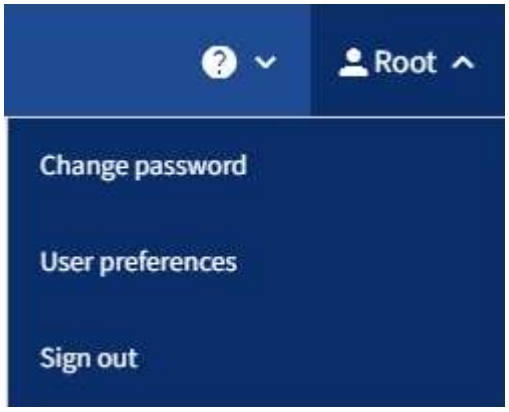
Sign out of Tenant Manager

When you are done working with the Tenant Manager, you must sign out to ensure that unauthorized users can't 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**.
   - If SSO is not in use:

     You are signed out of the Admin Node. The Tenant Manager sign in page is displayed.

     > If you signed into more than one Admin Node, you must sign out of each node.

   - If SSO is enabled:

     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.

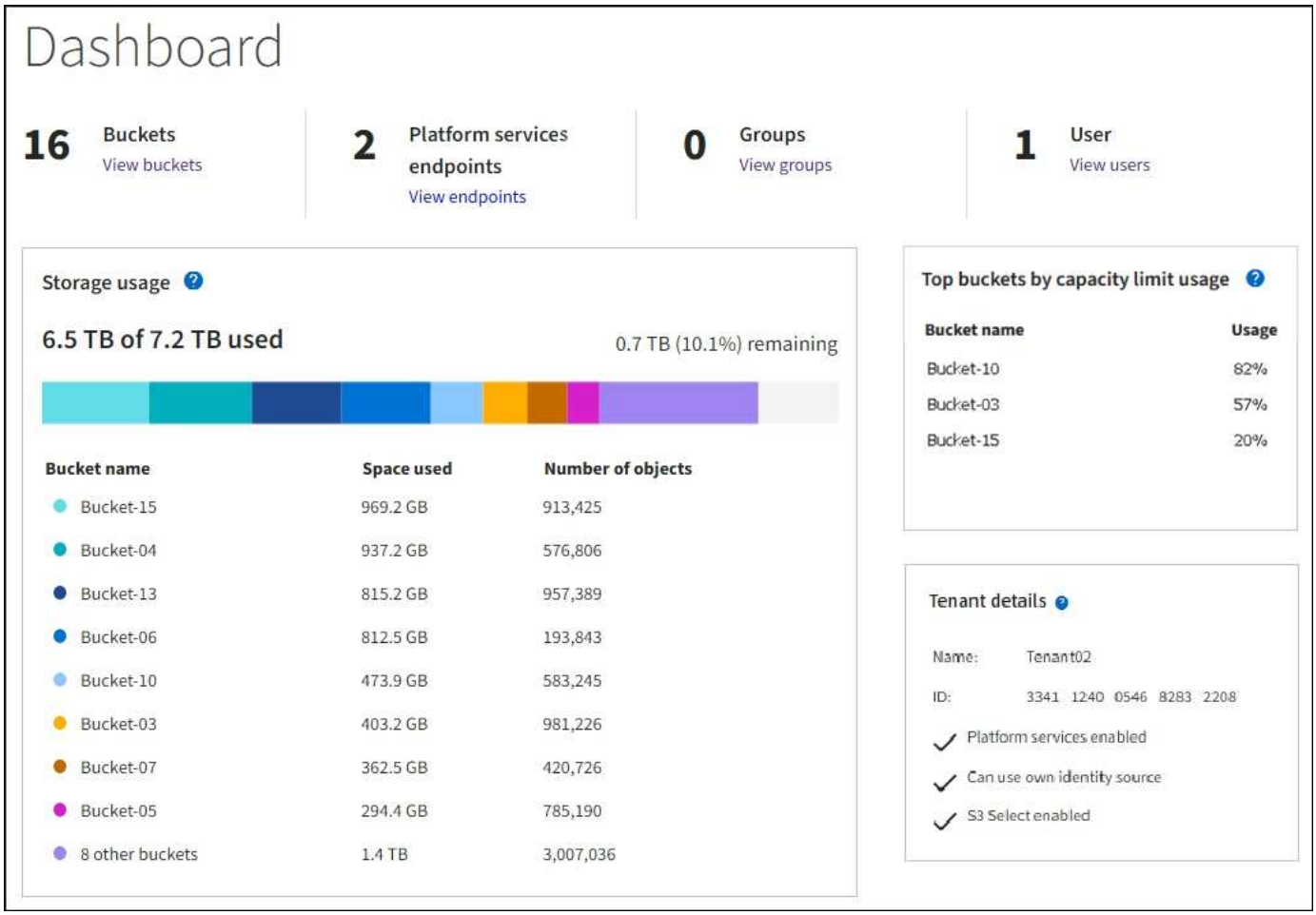
     > If SSO is enabled and you are also signed in to the Grid Manager, you must also sign out of the Grid Manager to sign out of SSO.

Understand 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:



Tenant account information

The top of the dashboard displays the number of configured buckets or containers, groups, and users. It also displays the number of platform services endpoints, if any have been configured. Select the links to view the details.

Depending on the tenant management permissions you have and the options you've configured, the remainder of the dashboard displays various combinations of guidelines, storage usage, object information, and tenant details.

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 usage 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 usage is recalculated. Quota usage 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.

To change units for the storage values displayed in the Tenant Manager, select the user drop-down in the upper right of the Tenant Manager, then select **User preferences**.

Quota usage alerts

If quota usage alerts have been enabled in the Grid Manager, these alerts 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.

  Consider asking your grid administrator to increase the quota.

- If you exceed your quota, a notification tells you that you can't upload new objects.

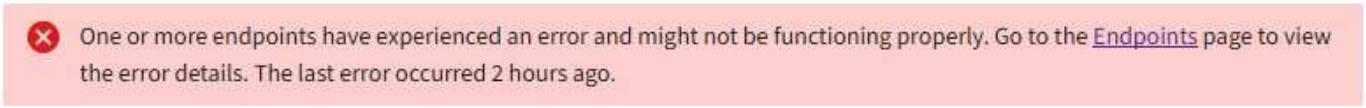
Capacity limit usage

If you've set a capacity limit for your buckets, the Tenant Manager dashboard displays a list of top buckets by capacity limit usage.

If no limit is set for a bucket, its capacity is unlimited. However, if your tenant account has a total storage quota and that quota is reached, you won't be able to ingest more objects regardless of the remaining capacity limit on a bucket.

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.



To see details about platform services endpoint errors, select **Endpoints** to display the Endpoints page.

Tenant Management API

Understand 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.
- Uses versioning to support non-disruptive upgrades.

To access the Swagger documentation for the Tenant Management API:

1. Sign in to the Tenant Manager.
2. From the top of the Tenant Manager, select the help icon and select **API documentation**.

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").

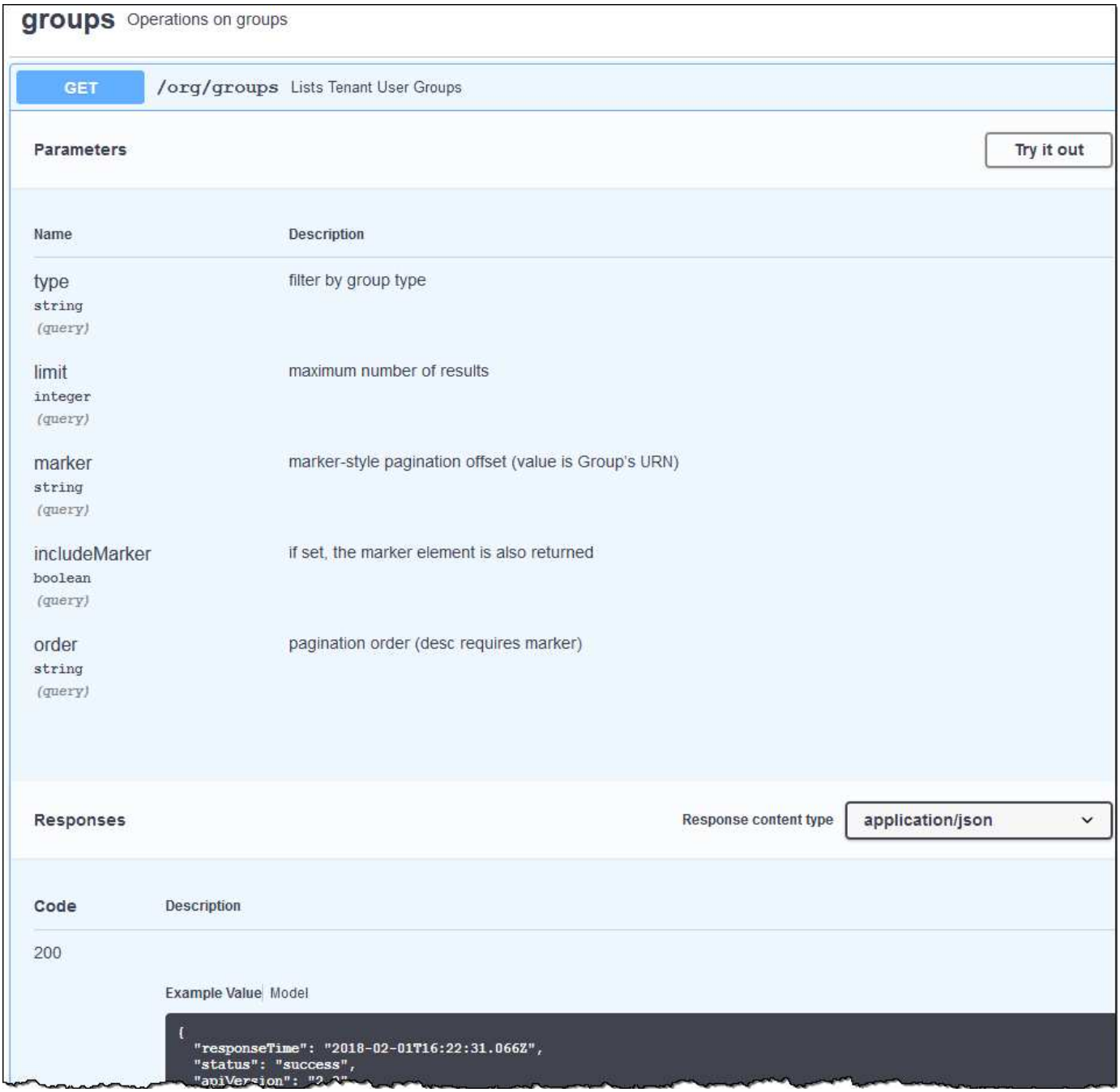
For information about improving authentication security, see Protect against Cross-Site Request Forgery.

> If single sign-on (SSO) is enabled for the StorageGRID system, you must perform different steps to authenticate. See the instructions for using the Grid Management API.

- **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.
- **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.
- **grid-federation-connections**: Operations on grid federation connections and cross-grid replication.
- **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.
- **ilm**: Operations on information lifecycle management (ILM) settings.
- **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 on global S3 Object Lock settings, used to support regulatory compliance.
- **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.



Issue API requests

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

Steps

1. Select 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 select **Model** to learn the requirements for each field.
4. Select **Try it out**.

5. Provide any required parameters, or modify the request body as required.
6. Select **Execute**.
7. Review the response code to determine if the request was successful.

Tenant Management API versioning

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

For example, this Request URL specifies version 4 of the API.

`https://hostname_or_ip_address/api/v4/authorize`

The major version of the API is bumped when changes are made that are *not compatible* with older versions. The minor version of the 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
	3.0	4.0

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

> You can configure the supported versions. See the **config** section of the Swagger API documentation for the Grid Management API for more information. You should deactivate support for the older version after updating all API clients to use the newer version.

Outdated requests are marked as deprecated in the following ways:

- The response header is "Deprecated: true"
- The JSON response body includes "deprecated": true
- A deprecated warning is added to nms.log. For example:

```
Received call to deprecated v2 API at POST "/api/v2/authorize"
```

Determine which API versions are supported in the current release

Use the `GET /versions` API request to return a list of the supported API major versions. This request is located in the **config** section of the Swagger API documentation.

```
GET https://{{IP-Address}}/api/versions
{
  "responseTime": "2023-06-27T22:13:50.750Z",
  "status": "success",
  "apiVersion": "4.0",
  "data": [
    2,
    3,
    4
  ]
}
```

Specify an API version for a request

You can specify the API version using a path parameter (`/api/v4`) or a header (`Api-Version: 4`). If you provide both values, the header value overrides the path value.

```
curl https://[IP-Address]/api/v4/grid/accounts

curl -H "Api-Version: 4" https://[IP-Address]/api/grid/accounts
```

Protect 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.

To configure CSRF protection, use the Grid Management API or Tenant Management API.

> 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.

Use grid federation connections

Clone tenant groups and users

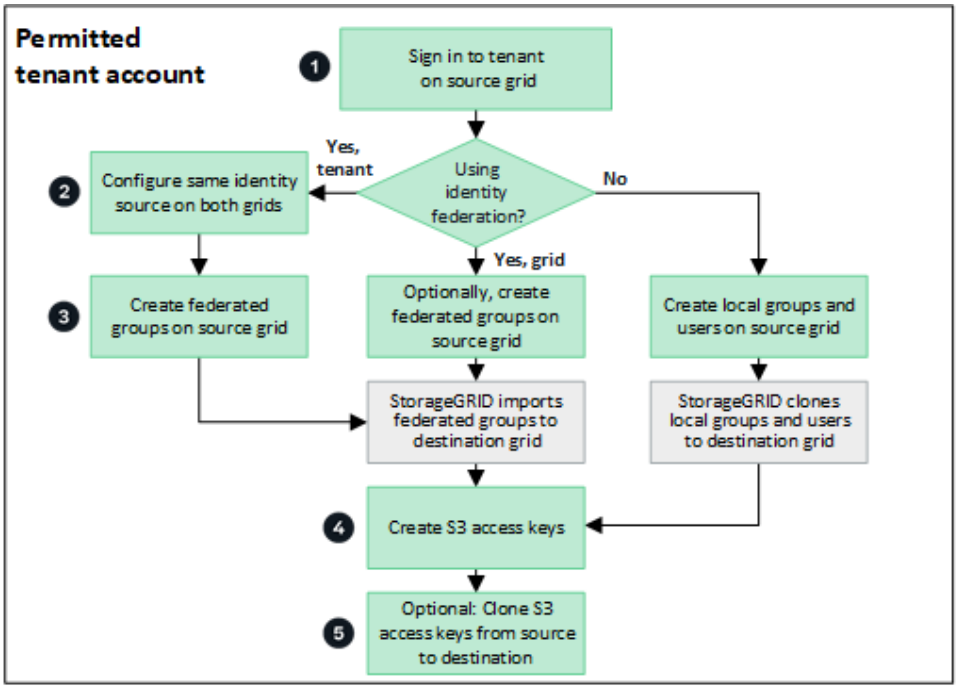
If a tenant was created or edited to use a grid federation connection, that tenant is replicated from one StorageGRID system (the source tenant) to another StorageGRID system (the replica tenant). After the tenant has been replicated, any groups and users added to the source tenant are cloned to the replica tenant.

The StorageGRID system where the tenant is originally created is the tenant's *source grid*. The StorageGRID system where the tenant is replicated is the tenant's *destination grid*. Both tenant accounts have the same account ID, name, description, storage quota, and assigned permissions, but the destination tenant does not initially have a root user password. For details, see What is account clone and Manage permitted tenants.

The cloning of tenant account information is required for cross-grid replication of bucket objects. Having the same tenant groups and users on both grids ensures you can access the corresponding buckets and objects on either grid.

Tenant workflow for account clone

If your tenant account has the **Use grid federation connection** permission, review the workflow diagram to see the steps you will perform to clone groups, users, and S3 access keys.



These are the primary steps in the workflow:

1 Sign in to tenant

Sign in to the tenant account on the source grid (the grid where the tenant was initially created.)

2 Optionally, configure identity federation

If your tenant account has the **Use own identity source** permission to use federated groups and users, configure the same identity source (with the same settings) for both the source and destination tenant accounts. Federated groups and users can't be cloned unless both grids are using the same identity source. For instructions, see Use identity federation.

3 Create groups and users

When creating groups and users, always start from the tenant's source grid. When you add a new group, StorageGRID automatically clones it to the destination grid.

- If identity federation is configured for the entire StorageGRID system or for your tenant account, create new tenant groups by importing federated groups from the identity source.
- If you aren't using identity federation, create new local groups and then create local users.

4 Create S3 access keys

You can create your own access keys or to create another user's access keys on either the source grid or the destination grid to access buckets on that grid.

5 Optionally, clone S3 access keys

If you need to access buckets with the same access keys on both grids, create the access keys on the source grid and then use the Tenant Manager API to manually clone them to the destination grid. For instructions, see Clone S3 access keys using the API.

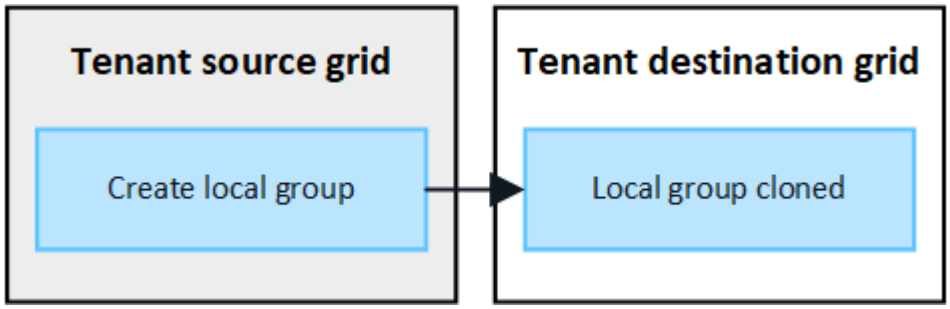
How are groups, users, and S3 access keys cloned?

Review this section to understand how groups, users, and S3 access keys are cloned between the tenant source grid and the tenant destination grid.

Local groups created on source grid are cloned

After a tenant account is created and replicated to the destination grid, StorageGRID automatically clones any local groups you add to the tenant's source grid to the tenant's destination grid.

Both the original group and its clone have the same access mode, group permissions, and S3 group policy. For instructions, see Create groups for S3 tenant.

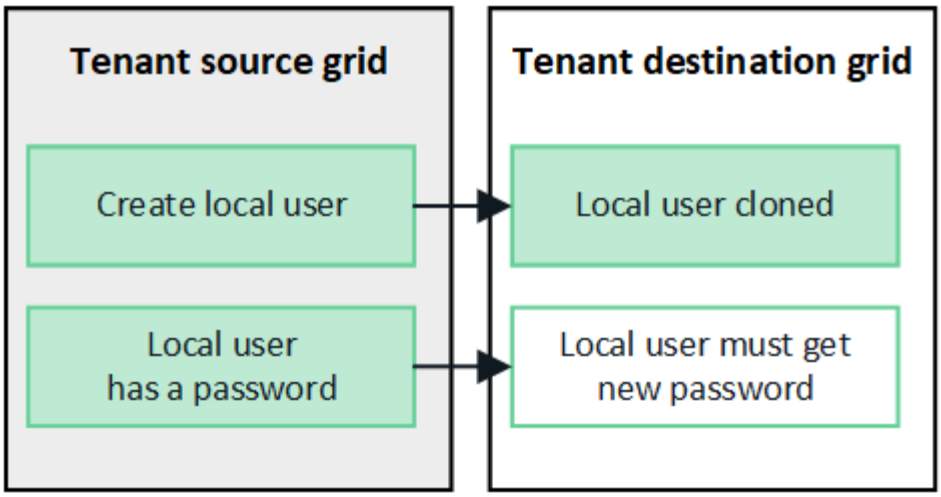


Any users you select when you create a local group on the source grid aren't included when the group is cloned to the destination grid. For this reason, don't select users when you create the group. Instead, select the group when you create the users.

Local users created on source grid are cloned

When you create a new local user on the source grid, StorageGRID automatically clones that user to the destination grid. Both the original user and its clone have the same full name, username, and **Deny access** setting. Both users also belong to the same groups. For instructions, see Manage local users.

For security reasons, local user passwords aren't cloned to the destination grid. If a local user needs to access Tenant Manager on the destination grid, the root user for the tenant account must add a password for that user on the destination grid. For instructions, see Manage local users.

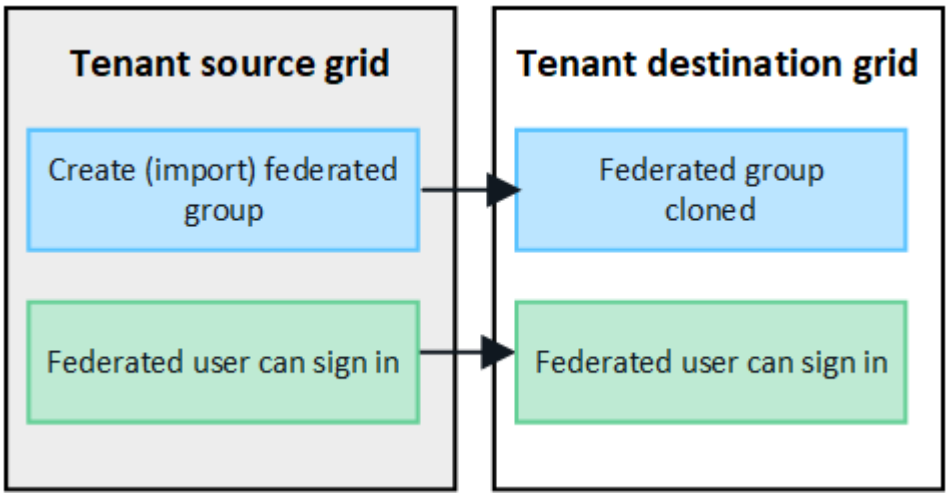


Federated groups created on source grid are cloned

Assuming the requirements for using account clone with single sign-on and identity federation have been met, federated groups that you create (import) for the tenant on the source grid are automatically cloned to the tenant on the destination grid.

Both groups have the same access mode, group permissions and S3 group policy.

After federated groups are created for the source tenant and cloned to the destination tenant, federated users can sign in to the tenant on either grid.

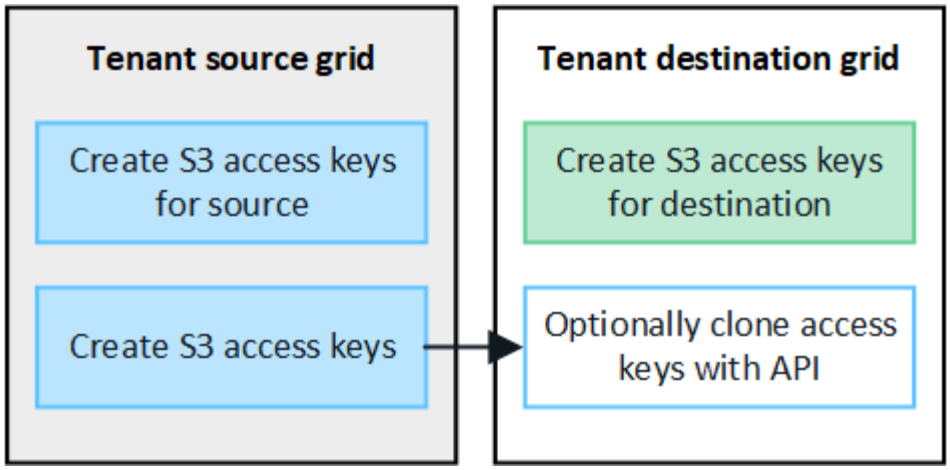


S3 access keys can be manually cloned

StorageGRID does not automatically clone S3 access keys because security is improved by having different keys on each grid.

To manage access keys on the two grids, you can do either of the following:

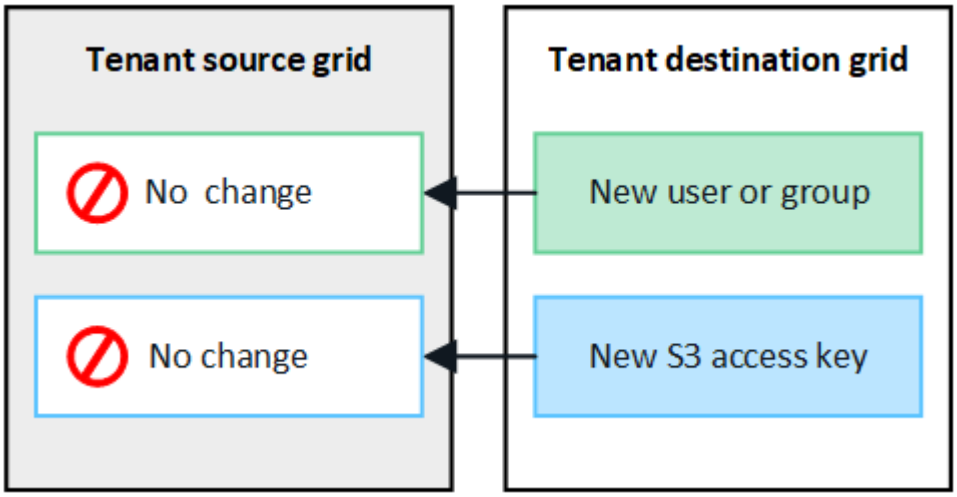
- If you don’t need to use the same keys for each grid, you can create your own access keys or create another user’s access keys on each grid.
- If you need to use the same keys on both grids, you can create keys on the source grid and then use the Tenant Manager API to manually clone the keys to the destination grid.



When you clone S3 access keys for a federated user, both the user and the S3 access keys are cloned to the destination tenant.

Groups and users added to destination grid aren't cloned

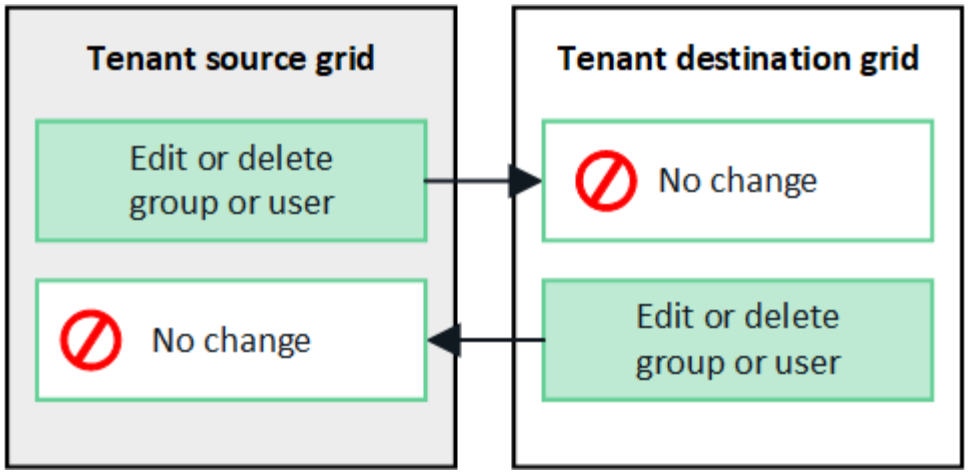
Cloning occurs only from the tenant's source grid to the tenant's destination grid. If you create or import groups and users on the tenant's destination grid, StorageGRID will not clone these items back the tenant's source grid.



Edited or deleted groups, users, and access keys aren't cloned

Cloning occurs only when you create new groups and users.

If you edit or delete groups, users, or access keys on either grid, your changes will not be cloned to the other grid.



Clone S3 access keys using the API

If your tenant account has the **Use grid federation connection** permission, you can use the Tenant Management API to manually clone S3 access keys from the tenant on the source grid to the tenant on the destination grid.

Before you begin

- The tenant account has the **Use grid federation connection** permission.
- The grid federation connection has a **Connection status** of **Connected**.
- You are signed in to the Tenant Manager on the tenant's source grid using a supported web browser.
- You belong to a user group that has the Manage your own S3 credentials or Root access permission.
- If you are cloning access keys for a local user, the user already exists on both grids.

> When you clone S3 access keys for a federated user, both the user and the S3 access keys are added to the destination tenant.

Clone your own access keys

You can clone your own access keys if you need to access the same buckets on both grids.

Steps

1. Using the Tenant Manager on the source grid, create your own access keys and download the `.csv` file.
2. From the top of the Tenant Manager, select the help icon and select **API documentation**.
3. In the **s3** section, select the following endpoint:

   ```
   POST /org/users/current-user/replicate-s3-access-key
   ```

4. Select **Try it out**.
5. In the **body** text box, replace the example entries for **accessKey** and **secretAccessKey** with the values from the **.csv** file you downloaded.

   Be sure to retain the double quotes around each string.

6. If the key will expire, replace the example entry for **expires** with the expiration date and time as a string in ISO 8601 data-time format (for example, `2024-02-28T22:46:33-08:00`). If the key will not expire, enter **null** as the value for the **expires** entry (or remove the **Expires** line and the preceding comma).
7. Select **Execute**.
8. Confirm that the server response code is **204**, indicating that the key was successfully cloned to the destination grid.

Clone another user's access keys

You can clone another user's access keys if they need to access the same buckets on both grids.

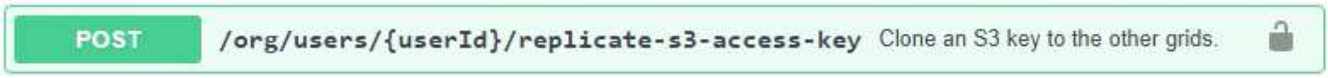
Steps

1. Using the Tenant Manager on the source grid, create the other user's S3 access keys and download the `.csv` file.
2. From the top of the Tenant Manager, select the help icon and select **API documentation**.
3. Obtain the user ID. You will need this value to clone the other user's access keys.
   a. From the **users** section, select the following endpoint:

      `GET /org/users`

   b. Select **Try it out**.
   c. Specify any parameters you want to use when looking up users.
   d. Select **Execute**.
   e. Find the user whose keys you want to clone, and copy the number in the **id** field.
4. In the **s3** section, select the following endpoint:

   `POST /org/users/{userId}/replicate-s3-access-key`



5. Select **Try it out**.
6. In the **userId** text box, paste the user ID you copied.
7. In the **body** text box, replace the example entries for **example access key** and **secret access key** with the values from the **.csv** file for that user.

   Be sure to retain the double quotes around the string.

8. If the key will expire, replace the example entry for **expires** with the expiration date and time as a string in

ISO 8601 data-time format (for example, `2023-02-28T22:46:33-08:00`). If the key will not expire, enter **null** as the value for the **expires** entry (or remove the **Expires** line and the preceding comma).

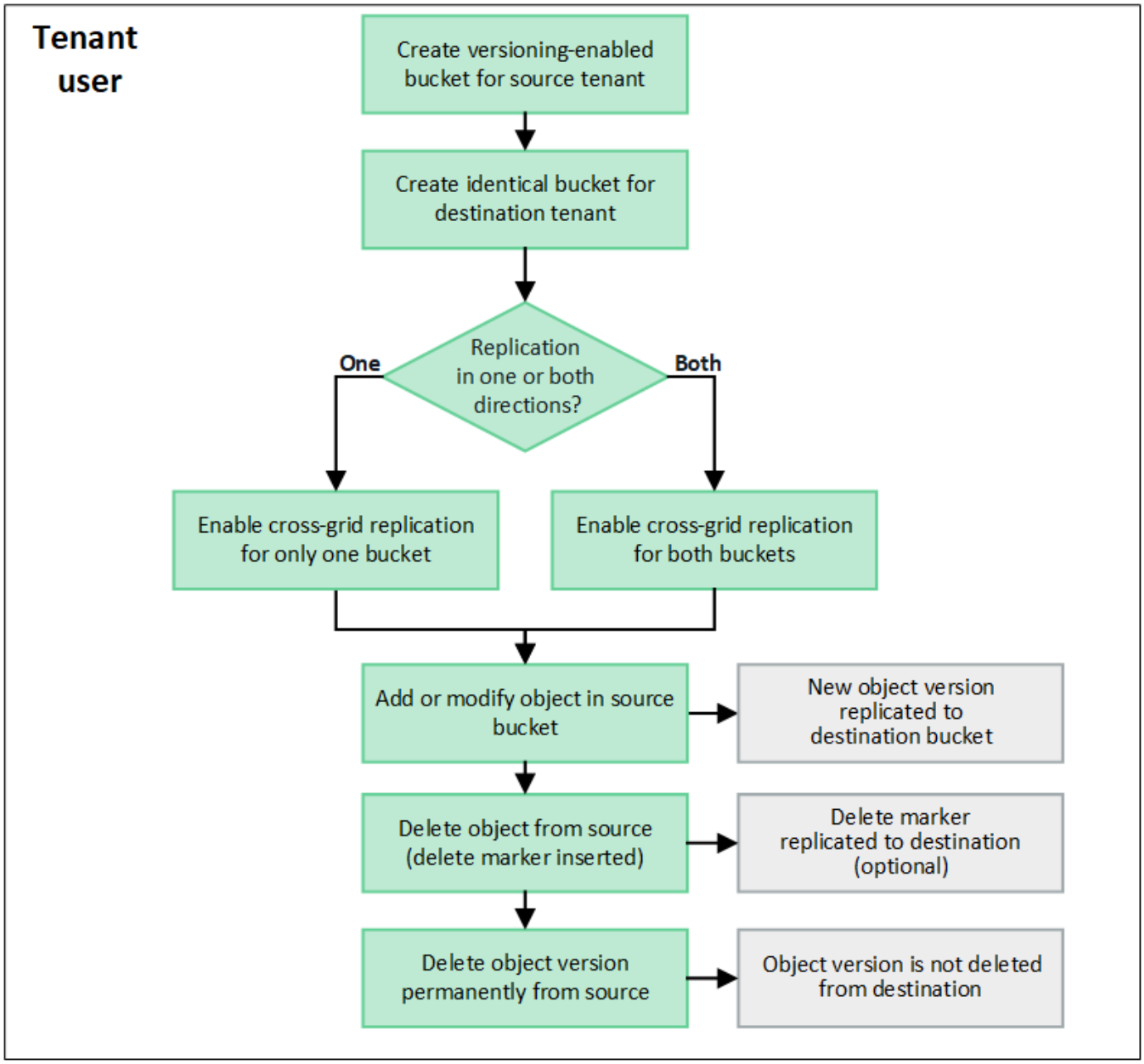
9. Select **Execute**.
10. Confirm that the server response code is **204**, indicating that the key was successfully cloned to the destination grid.

Manage cross-grid replication

If your tenant account was assigned the **Use grid federation connection** permission when it was created, you can use cross-grid replication to automatically replicate objects between buckets on the tenant's source grid and buckets on the tenant's destination grid. Cross-grid replication can occur in one or both directions.

Workflow for cross-grid replication

The workflow diagram summarize the steps you will perform to configure cross-grid replication between buckets on two grids. These steps are described in more detail below.



Configure cross-grid replication

Before you can use cross-grid replication, you must sign in to the corresponding tenant accounts on each grid and create identical buckets. Then, you can enable cross-grid replication on either or both buckets.

Before you begin

- You have reviewed the requirements for cross-grid replication. See What is cross-grid replication.
- You are using a supported web browser.
- The tenant account has the **Use grid federation connection** permission, and identical tenant accounts exist on both grids. See Manage the permitted tenants for grid federation connection.
- The tenant user you will be signing in as already exists on both grids and belongs to a user group that has the Root access permission.
- If you will be signing in to the tenant's destination grid as a local user, the root user for the tenant account has set a password for your user account on that grid.

Create two identical buckets

As a first step, sign in to the corresponding tenant accounts on each grid and create identical buckets.

Steps

1. Starting from either grid in the grid federation connection, create a new bucket:
   a. Sign in to the tenant account using the credentials of a tenant user who exists on both grids.

      If you are unable to sign in to the tenant's destination grid as a local user, confirm that the root user for the tenant account has set a password for your user account.

   b. Follow the instructions to create an S3 bucket.
   c. On the **Manage object settings** tab, select **Enable object versioning**.
   d. If S3 Object Lock is enabled for your StorageGRID system, don't enable S3 Object Lock for the bucket.
   e. Select **Create bucket**.
   f. Select **Finish**.
2. Repeat these steps to create an identical bucket for the same tenant account on the other grid in the grid federation connection.

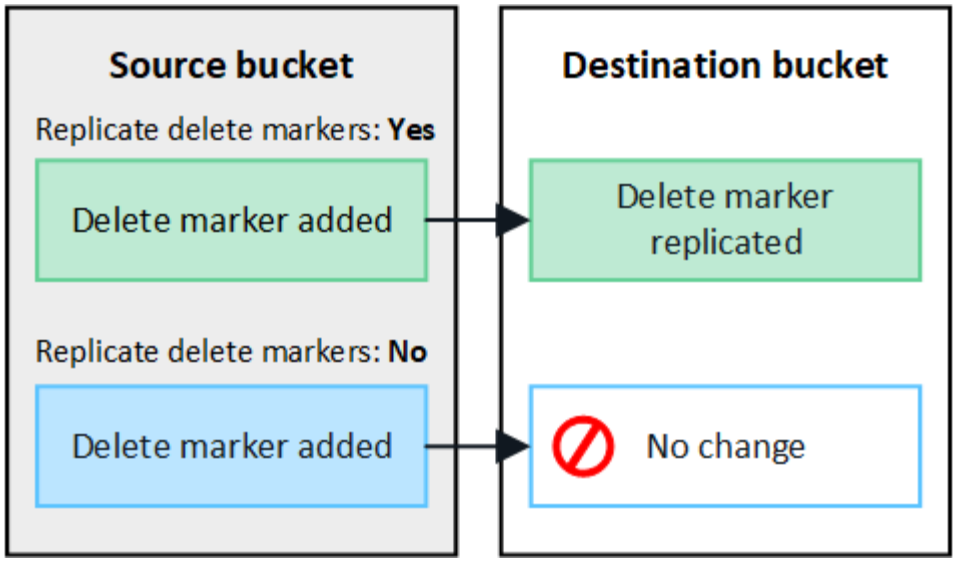
   As required, each bucket can use a different region.

Enable cross-grid replication

You must perform these steps before adding any objects to either bucket.

Steps

1. Starting from a grid whose objects you want to replicate, enable cross-grid replication in one direction:
   a. Sign in to the tenant account for the bucket.
   b. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.
   c. Select the bucket name from the table to access the bucket details page.
   d. Select the **Cross-grid replication** tab.
   e. Select **Enable**, and review the list of requirements.
   f. If all requirements have been met, select the grid federation connection you want to use.
   g. Optionally, change the setting of **Replicate delete markers** to determine what happens on the destination grid if an S3 client issues a delete request to the source grid that doesn't include a version ID:
      - **Yes** (default): A delete marker is added to the source bucket and replicated to the destination bucket.
      - **No**: A delete marker is added to the source bucket but is not replicated to the destination bucket.



If the delete request includes a version ID, that object version is permanently removed from the source bucket. StorageGRID does not replicate delete requests that include a version ID, so the same object version is not deleted from the destination.

See What is cross-grid replication for details.

h. Optionally, change the setting of the **Cross-grid replication** audit category to manage the volume of audit messages:

- **Error** (default): Only failed cross-grid replication requests are included in the audit output.
- **Normal**: All cross-grid replication requests are included, which significantly increases the volume of the audit output.

i. Review your selections. You aren't able to change these settings unless both buckets are empty.

j. Select **Enable and test**.

After a few moments, a success message appears. Objects added to this bucket will now be automatically replicated to the other grid. **Cross-grid replication** is shown as an enabled feature on the bucket details page.

2. Optionally, go to the corresponding bucket on the other grid and enable cross-grid replication in both directions.

Test replication between grids

If cross-grid replication is enabled for a bucket, you might need to verify that the connection and cross-grid replication are working correctly and that the source and destination buckets still meet all requirements (for example, versioning is still enabled).

Before you begin

- You are using a supported web browser.
- You belong to a user group that has the Root access permission.

Steps

1. Sign in to the tenant account for the bucket.

2. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.
3. Select the bucket name from the table to access the bucket details page.
4. Select the **Cross-grid replication** tab.
5. Select **Test connection**.

   If the connection is healthy, a success banner appears. Otherwise, an error message appears, which you and the grid admin can use to resolve the issue. For details, see Troubleshoot grid federation errors.

6. If cross-grid replication is configured to occur in both directions, go to the corresponding bucket on the other grid and select **Test connection** to verify that cross-grid replication is working in the other direction.

Disable cross-grid replication

You can permanently stop cross-grid replication if you no longer want to copy objects to the other grid.

Before disabling cross-grid replication, note the following:

- Disabling cross-grid replication does not remove any objects that have already been copied between grids. For example, objects in `my-bucket` on Grid 1 that have been copied to `my-bucket` on Grid 2 aren't removed if you disable cross-grid replication for that bucket. If you want to delete these objects, you must remove them manually.
- If cross-grid replication was enabled for each of the buckets (that is, if replication occurs in both directions), you can disable cross-grid replication for either or both buckets. For example, you might want to disable replicating objects from `my-bucket` on Grid 1 to `my-bucket` on Grid 2, while continuing to replicate objects from `my-bucket` on Grid 2 to `my-bucket` on Grid 1.
- You must disable cross-grid replication before you can remove a tenant's permission to use the grid federation connection. See Manage permitted tenants.
- If you disable cross-grid replication for a bucket that contains objects, you will not be able to reenable cross-grid replication unless you delete all objects from both the source and destination buckets.

> ⚠ You can't reenable replication unless both buckets are empty.

Before you begin

- You are using a supported web browser.
- You belong to a user group that has the Root access permission.

Steps

1. Starting from the grid whose objects you no longer want to replicate, stop cross-grid replication for the bucket:
   a. Sign in to the tenant account for the bucket.
   b. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.
   c. Select the bucket name from the table to access the bucket details page.
   d. Select the **Cross-grid replication** tab.
   e. Select **Disable replication**.
   f. If you are sure you want to disable cross-grid replication for this bucket, type **Yes** in the text box, and select **Disable**.

After a few moments, a success message appears. New objects added to this bucket can no longer be automatically replicated to the other grid. **Cross-grid replication** is no longer shown as a Enabled feature on the Buckets page.

2. If cross-grid replication was configured to occur in both directions, go to the corresponding bucket on the other grid and stop cross-grid replication in the other direction.

View grid federation connections

If your tenant account has the **Use grid federation connection** permission, you can view the allowed connections.

Before you begin

- The tenant account has the **Use grid federation connection** permission.
- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Root access permission.

Steps

1. Select **STORAGE (S3)** > **Grid federation connections**.

   The Grid federation connection page appears and includes a table that summarizes the following information:

Column	Description
Connection name	The grid federation connections this tenant has permission to use.
Buckets with cross-grid replication	For each grid federation connection, the tenant buckets that have cross-grid replication enabled. Objects added to these buckets will be replicated to the other grid in the connection.
Last error	For each grid federation connection, the most recent error to occur, if any, when data was being replicated to the other grid. See Clear the last error.

2. Optionally, select a bucket name to view bucket details.

Clear the last error

An error might appear in the **Last error** column for one of these reasons:

- The source object version was not found.
- The source bucket was not found.
- The destination bucket was deleted.
- The destination bucket was re-created by a different account.
- The destination bucket has versioning suspended.
- The destination bucket was re-created by the same account but is now unversioned.

This column only shows the last cross-grid replication error to occur; previous errors that might have occurred will not be shown.

Steps

1. If a message appears in the **Last error** column, view the message text.

   For example, this error indicates that the destination bucket for cross-grid replication was in an invalid state, possibly because versioning was suspended or S3 Object Lock was enabled.

2. Perform any recommended actions. For example, if versioning was suspended on the destination bucket for cross-grid replication, reenable versioning for that bucket.
3. Select the connection from the table.
4. Select **Clear error**.
5. Select **Yes** to clear the message and update the system's status.
6. Wait 5-6 minutes and then ingest a new object into the bucket. Confirm that the error message does not reappear.

   To ensure the error message is cleared, wait at least 5 minutes after the timestamp in the message before ingesting a new object.

7. To determine if any objects failed to be replicated because of the bucket error, see Identify and retry failed replication operations.

Manage groups and users

Use 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.

Configure identity federation for Tenant Manager

You can configure identity federation for the Tenant Manager if you want tenant groups and users to be managed in another system such as Active Directory, Azure Active Directory (Azure AD), OpenLDAP, or Oracle Directory Server.

Before you begin

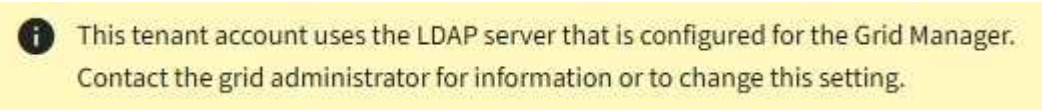
- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Root access permission.
- You are using Active Directory, Azure AD, OpenLDAP, or Oracle Directory Server as the identity provider.

  If you want to use an LDAP v3 service that is not listed, contact technical support.

- If you plan to use OpenLDAP, you must configure the OpenLDAP server. See Guidelines for configuring OpenLDAP server.
- 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. See Supported ciphers for outgoing TLS connections.

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 can't configure a separate federated identity source for this tenant.

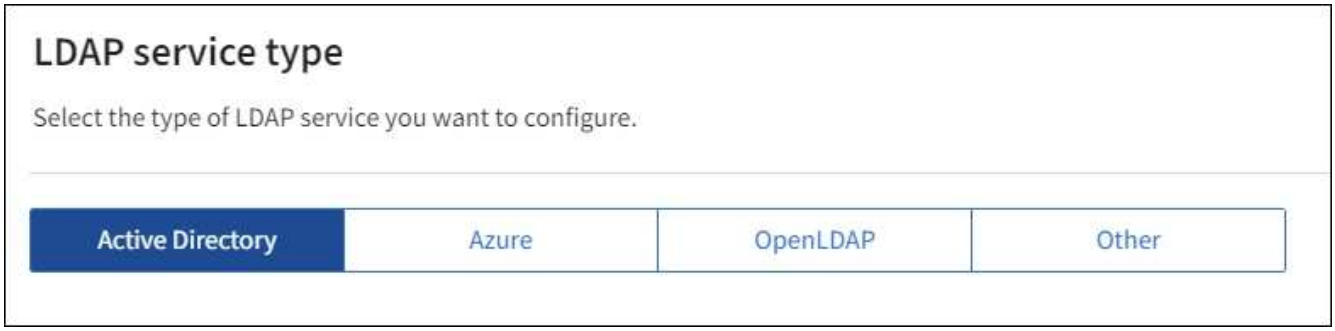


Enter configuration

When you configure identify federation, you provide the values StorageGRID needs to connect to an LDAP service.

Steps

1. Select **ACCESS MANAGEMENT** > **Identity federation**.
2. Select **Enable identity federation**.
3. In the LDAP service type section, select the type of LDAP service you want to configure.



   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. Otherwise, go to the next step.
   - **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. For all LDAP service types, enter the required LDAP server and network connection information in the Configure LDAP server section.
   - **Hostname**: The fully qualified domain name (FQDN) 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`
     - **Active Directory**: `objectSid`, `primaryGroupID`, `userAccountControl`, and `userPrincipalName`
     - **Azure**: `accountEnabled` and `userPrincipalName`

   - **Password**: The password associated with the username.

     If you change the password in the future, you must update it on this page.

   - **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.

- **Bind username format** (optional): The default username pattern StorageGRID should use if the pattern can't be determined automatically.

  Providing **Bind username format** is recommended because it can allow users to sign in if StorageGRID is unable to bind with the service account.

  Enter one of these patterns:

  - **UserPrincipalName pattern (Active Directory and Azure)**: `[USERNAME]@example.com`
  - **Down-level logon name pattern (Active Directory and Azure)**: `example\[USERNAME]`
  - **Distinguished name pattern**: `CN=[USERNAME],CN=Users,DC=example,DC=com`

    Include **[USERNAME]** exactly as written.

6. In the Transport Layer Security (TLS) section, select a security setting.
   - **Use STARTTLS**: Use STARTTLS to secure communications with the LDAP server. This is the recommended option for Active Directory, OpenLDAP, or Other, but this option is not supported for Azure.
   - **Use LDAPS**: The LDAPS (LDAP over SSL) option uses TLS to establish a connection to the LDAP server. You must select this option for Azure.
   - **Do not use TLS**: The network traffic between the StorageGRID system and the LDAP server will not be secured. This option is not supported for Azure.

     > Using the **Do not use TLS** 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 Grid 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.

Test the connection and save the configuration

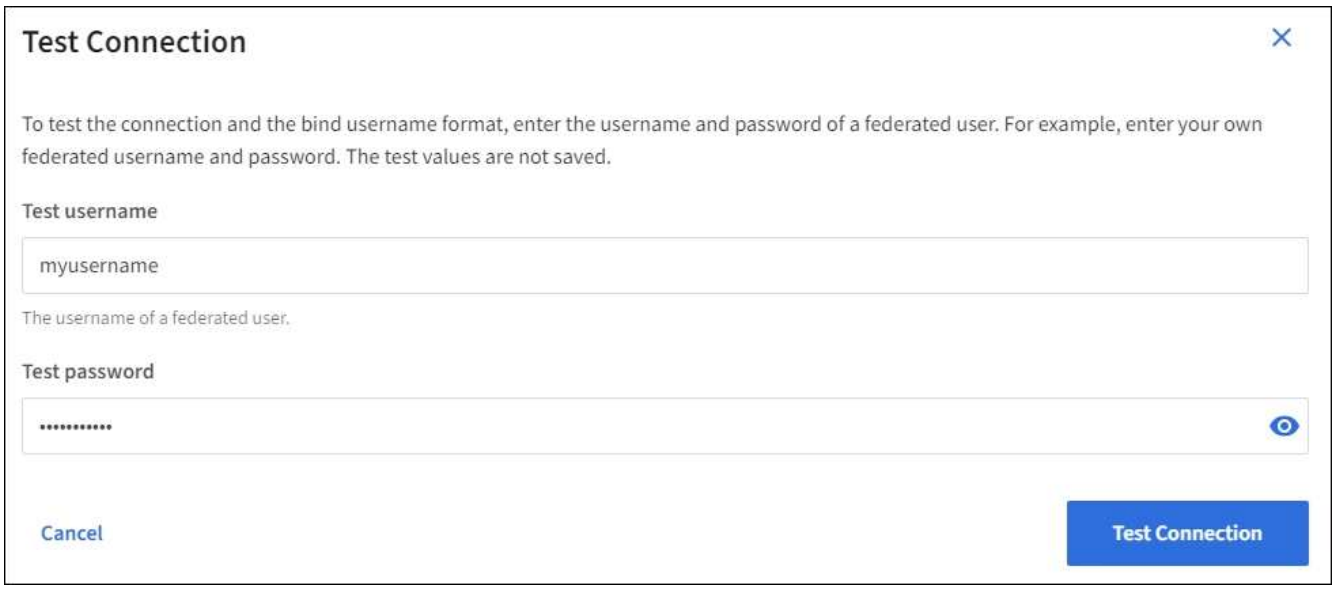
After entering all values, you must test the connection before you can save the configuration. StorageGRID verifies the connection settings for the LDAP server and the bind username format, if you provided one.

Steps

1. Select **Test connection**.
2. If you did not provide a bind username format:
   - A "Test connection successful" message appears if the connection settings are valid. Select **Save** to save the configuration.
   - A "test connection could not be established" message appears if the connection settings are invalid. Select **Close**. Then, resolve any issues and test the connection again.

3. If you provided a bind username format, enter the username and password of a valid federated user.

   For example, enter your own username and password. Don't include any special characters in the username, such as @ or /.



   - A "Test connection successful" message appears if the connection settings are valid. Select **Save** to save the configuration.
   - An error message appears if the connection settings, bind username format, or test username and password are invalid. Resolve any issues and test the connection again.

Force synchronization with 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.

Steps

1. Go to the Identity federation page.
2. Select **Sync server** at the top of the page.

   The synchronization process might take some time depending on your environment.

   > The **Identity federation synchronization failure** alert is triggered if there is an issue synchronizing federated groups and users from the identity source.

Disable identity federation

You can temporarily or permanently disable identity federation for groups and users. When identity federation is disabled, there is no communication between StorageGRID and the identity source. However, any settings you have configured are retained, allowing you to easily reenable identity federation in the future.

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 StorageGRID system 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, and alerts will not be raised for accounts that have not been synchronized.
- The **Enable identity federation** checkbox is disabled if single sign-on (SSO) is set to **Enabled** or **Sandbox Mode**. The SSO Status on the Single Sign-on page must be **Disabled** before you can disable identity federation. See Disable single sign-on.

Steps

1. Go to the Identity federation page.
2. Uncheck the **Enable identity federation** checkbox.

Guidelines for configuring OpenLDAP server

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

> For identity sources that aren't ActiveDirectory or Azure, StorageGRID will not automatically block S3 access to users who are disabled externally. To block S3 access, delete any S3 keys for the user or remove the user from all groups.

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 OpenLDAP documentation: Version 2.4 Administrator's Guide.

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 OpenLDAP documentation: Version 2.4 Administrator's Guide.

Manage tenant groups

Create groups for an S3 tenant

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

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You 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.
- If your tenant account has the **Use grid federation connection** permission, you have reviewed the workflow and considerations for cloning tenant groups and users, and you are signed in to the tenant's source grid.

Access the Create group wizard

As your first step, access the Create group wizard.

Steps

1. Select **ACCESS MANAGEMENT** > **Groups**.
2. If your tenant account has the **Use grid federation connection** permission, confirm that a blue banner appears, indicating that new groups created on this grid will be cloned to the same tenant on the other grid in the connection. If this banner does not appear, you might be signed in to the tenant's destination grid.

3. Select **Create group**.

Choose a group type

You can create a local group or import a federated group.

Steps

1. 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.

2. Enter the group's name.
   - **Local group**: Enter both a display name and a unique name. You can edit the display name later.

> If your tenant account has the **Use grid federation connection** permission, a cloning error will occur if the same **Unique name** already exists for the tenant on the destination grid.

- **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.

3. Select **Continue**.

Manage group permissions

Group permissions control which tasks users can perform in the Tenant Manager and Tenant Management API.

Steps

1. For **Access mode**, select one of the following:
   - **Read-write** (default): Users can sign in to Tenant Manager and manage the tenant configuration.
   - **Read-only**: Users can only view settings and features. They can't make any changes or perform any operations in the Tenant Manager or Tenant Management API. Local read-only users can change their own passwords.

   > 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.

2. Select one or more permissions for this group.

   See Tenant management permissions.

3. Select **Continue**.

Set S3 group policy

The group policy determines which S3 access permissions users will have.

Steps

1. Select the policy you want to use for this group.

Group policy	Description
No S3 Access	Default. Users in this group don't 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 can't edit this string.

Group policy	Description
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 can't edit this string.
Ransomware Mitigation	This example policy applies to all buckets for this tenant. Users in this group can perform common actions, but can't permanently delete objects from buckets that have object versioning enabled. Tenant Manager users who have the **Manage all buckets** permission can override this group policy. Limit the Manage all buckets permission to trusted users, and use Multi-Factor Authentication (MFA) where available.
Custom	Users in the group are granted the permissions you specify in the text box.

2. 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.

   For detailed information about group policies, including language syntax and examples, see Example group policies.

3. If you are creating a local group, select **Continue**. If you are creating a federated group, select **Create group** and **Finish**.

Add users (local groups only)

You can save the group without adding users, or you can optionally add any local users that already exist.

> If your tenant account has the **Use grid federation connection** permission, any users you select when you create a local group on the source grid aren't included when the group is cloned to the destination grid. For this reason, don't select users when you create the group. Instead, select the group when you create the users.

Steps

1. Optionally, select one or more local users for this group.
2. Select **Create group** and **Finish**.

   The group you created appears in the list of groups.

   If your tenant account has the **Use grid federation connection** permission and you are on the tenant's source grid, the new group is cloned to the tenant's destination grid. **Success** appears as the **Cloning status** in the Overview section of the group's detail page.

Create 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.

> Support for Swift client applications has been deprecated and will be removed in a future release.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You 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.

Access the Create group wizard

Steps

As your first step, access the Create group wizard.

1. Select **ACCESS MANAGEMENT** > **Groups**.
2. Select **Create group**.

Choose a group type

You can create a local group or import a federated group.

Steps

1. 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.

2. 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.
3. Select **Continue**.

Manage group permissions

Group permissions control which tasks users can perform in the Tenant Manager and Tenant Management API.

Steps

1. For **Access mode**, select one of the following:
   - **Read-write** (default): Users can sign in to Tenant Manager and manage the tenant configuration.
   - **Read-only**: Users can only view settings and features. They can't make any changes or perform any operations in the Tenant Manager or Tenant Management API. Local read-only users can change their own passwords.

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.

2. Select the **Root access** checkbox if group users need to sign in to the Tenant Manager or Tenant Management API.
3. Select **Continue**.

Set Swift group policy

Swift users need administrator permission to authenticate into the Swift REST API to create containers and ingest objects.

1. Select the **Swift administrator** checkbox if group users need to use the Swift REST API to manage containers and objects.
2. If you are creating a local group, select **Continue**. If you are creating a federated group, select **Create group** and **Finish**.

Add users (local groups only)

You can save the group without adding users, or you can optionally add any local users that already exist.

Steps

1. Optionally, select one or more local users for this group.

   If you have not yet created local users, you can add this group to the user on the Users page. See Manage local users.

2. Select **Create group** and **Finish**.

   The group you created appears in the list of groups.

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.

Permission	Description	Details
Root access	Provides full access to the Tenant Manager and the Tenant Management API.	Swift users must have Root access permission to sign in to the tenant account.
Administrator	Swift tenants only. Provides full access to the Swift containers and objects for this tenant account	Swift users must have the Swift Administrator permission to perform any operations with the Swift REST API.
Manage your own S3 credentials	Allows users to create and remove their own S3 access keys.	Users who don't have this permission don't see the **STORAGE (S3)** > **My S3 access keys** menu option.
View all buckets	**S3 tenants**: Allows users to view all buckets and bucket configurations. **Swift tenants**: Allows Swift users to view all containers and container configurations using the Tenant Management API.	Users who don't have either the View all buckets or the Manage all buckets permission don't see the **Buckets** menu option. This permission is superseded by the Manage all buckets permission. It does not affect S3 bucket or group polices used by S3 clients or S3 Console. You can only assign this permission to Swift groups from the Tenant Management API. You can't assign this permission to Swift groups using the Tenant Manager.
Manage all buckets	**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. **Swift tenants**: Allows Swift users to control the consistency for Swift containers using the Tenant Management API.	Users who don't have either the View all buckets or the Manage all buckets permission don't see the **Buckets** menu option. This permission supersedes the View all buckets permission. It does not affect S3 bucket or group polices used by S3 clients or S3 Console. You can only assign this permission to Swift groups from the Tenant Management API. You can't assign this permission to Swift groups using the Tenant Manager.
Manage endpoints	Allows users to use the Tenant Manager or the Tenant Management API to create or edit platform service endpoints, which are used as the destination for StorageGRID platform services.	Users who don't have this permission don't see the **Platform services endpoints** menu option.

Permission	Description	Details
Use S3 Console tab	When combined with the View all buckets or Manage all buckets permission, allows users to view and manage objects from the S3 Console tab on the details page for a bucket.	

Manage groups

Manage your tenant groups as needed to view, edit, or duplicate a group, and more.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Root access permission.

View or edit group

You can view and edit the basic information and details for each group.

Steps

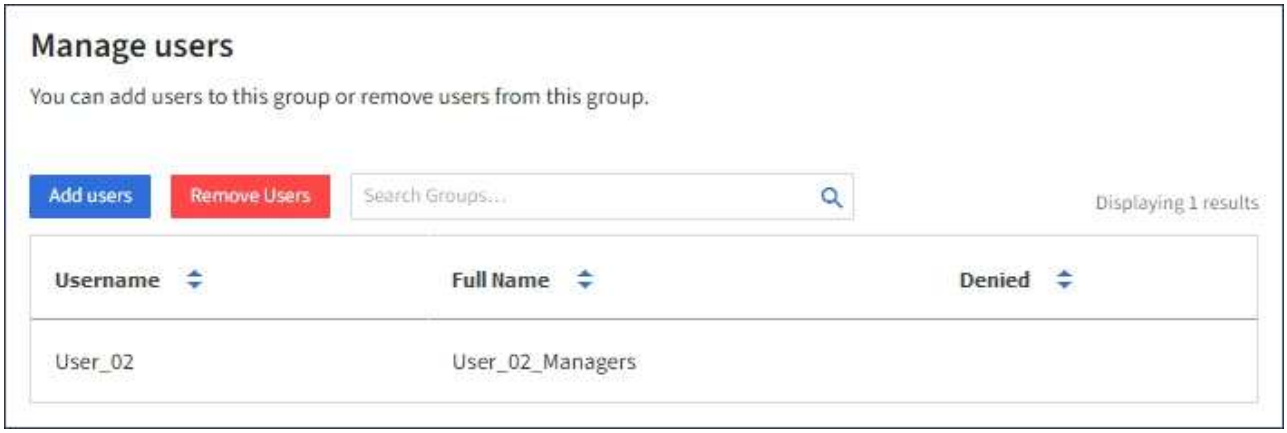
1. Select **ACCESS MANAGEMENT** > **Groups**.
2. Review the information provided on the Groups page, which lists basic information for all local and federated groups for this tenant account.

   If the tenant account has the **Use grid federation connection** permission and you are viewing groups on the tenant's source grid:

   - A banner message indicates that if you edit or remove a group, your changes will not be synced to the other grid.
   - As needed, a banner message indicates if groups were not cloned to the tenant on the destination grid. You can retry a group clone that failed.
3. If you want to change the group's name:
   a. Select the checkbox for the group.
   b. Select **Actions** > **Edit group name**.
   c. Enter the new name.
   d. Select **Save changes.**
4. If you want to view more details or make additional edits, do either of the following:
   - Select the group name.
   - Select the checkbox for the group, and select **Actions** > **View group details**.
5. Review the Overview section, which shows the following information for each group:
   - Display name
   - Unique name
   - Type
   - Access mode
   - Permissions

- S3 Policy
- Number of users in this group
- Additional fields if the tenant account has the **Use grid federation connection** permission and you are viewing the group on the tenant's source grid:
  - Cloning status, either **Success** or **Failure**
  - A blue banner indicating that if you edit or delete this group, your changes will not be synced to the other grid.

6. Edit group settings as needed. See Create groups for an S3 tenant and Create groups for a Swift tenant for details about what to enter.
   a. In the Overview section, change the display name by selecting the name or the edit icon .
   b. On the **Group permissions** tab, update the permissions, and select **Save changes**.
   c. On the **Group policy** tab, make any changes, and select **Save changes**.
      - If you are editing an S3 group, optionally select a different S3 group policy or enter the JSON string for a custom policy, as required.
      - If you are editing a Swift group, optionally select or clear the **Swift Administrator** checkbox.
7. To add one or more existing local users to the group:
   a. Select the Users tab.



   b. Select **Add users**.
   c. Select the existing users you want to add, and select **Add users**.

      A success message appears in the upper right.

8. To remove local users from the group:
   a. Select the Users tab.
   b. Select **Remove users**.
   c. Select the users you want to remove, and select **Remove users**.

      A success message appears in the upper right.

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

Duplicate group

You can duplicate an existing group to create new groups more quickly.

> If your tenant account has the **Use grid federation connection** permission and you duplicate a group from the tenant's source grid, the duplicated group will be cloned to the tenant's destination grid.

Steps

1. Select **ACCESS MANAGEMENT** > **Groups**.
2. Select the checkbox for the group you want to duplicate.
3. Select **Actions** > **Duplicate group**.
4. See Create groups for an S3 tenant or Create groups for a Swift tenant for details about what to enter.
5. Select **Create group**.

Retry group clone

To retry a clone that failed:

1. Select each group that indicates *(Cloning failed)* below the group name.
2. Select **Actions** > **Clone groups**.
3. View the status of the clone operation from the details page of each group you're cloning.

For additional information, see Clone tenant groups and users.

Delete one or more groups

You can delete one or more groups. Any users who belong only to a group that is deleted will no longer be able to sign in to the Tenant Manager or use the tenant account.

> If your tenant account has the **Use grid federation connection** permission and you delete a group, StorageGRID will not delete the corresponding group on the other grid. If you need to keep this information in sync, you must delete the same group from both grids.

Steps

1. Select **ACCESS MANAGEMENT** > **Groups**.
2. Select the checkbox for each group you want to delete.
3. Select **Actions** > **Delete group** or **Actions** > **Delete groups**.

   A confirmation dialog box appears.

4. Select **Delete group** or **Delete groups**.

Manage 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 can't remove the root user.

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 client applications to access the tenant's resources, based on group permissions.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Root access permission.
- If your tenant account has the **Use grid federation connection** permission, you have reviewed the workflow and considerations for cloning tenant groups and users, and you are signed in to the tenant's source grid.

Create a local user

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

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

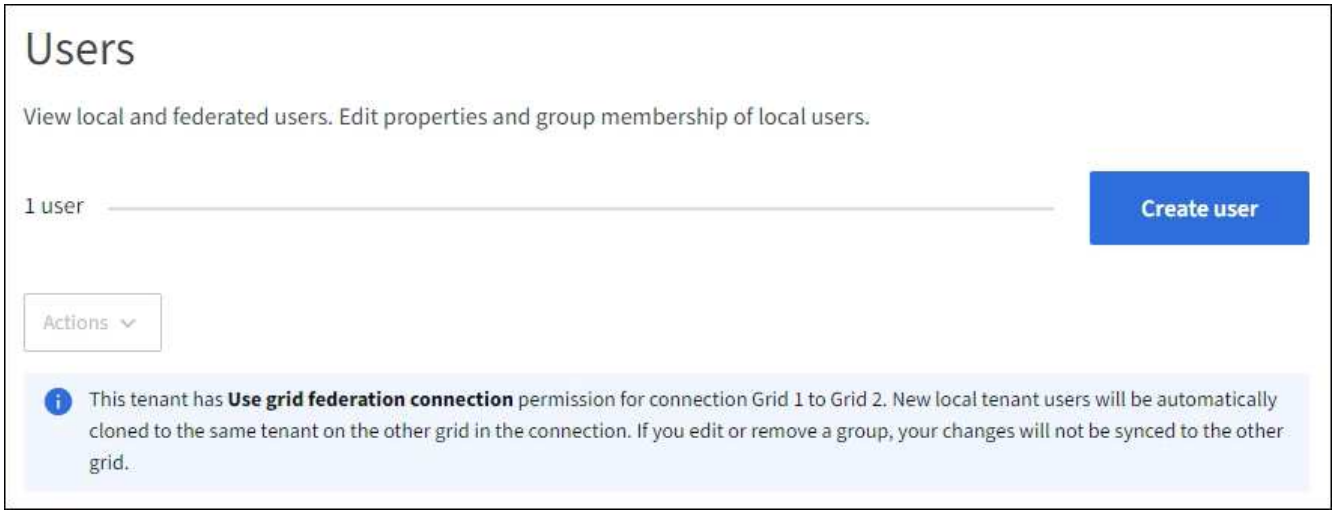
Swift users who don't belong to any groups don't have management permissions or Swift container access.

Access the Create user wizard

Steps

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

   If your tenant account has the **Use grid federation connection** permission, a blue banner indicates that this is the tenant's source grid. Any local users you create on this grid will be cloned to the other grid in the connection.



2. Select **Create user**.

Enter credentials

Steps

1. For the **Enter user credentials** step, complete the following fields.

Field	Description
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 can't be changed. **Note**: If your tenant account has the **Use grid federation connection** permission, a cloning error will occur if the same **Username** already exists for the tenant on the destination grid.
Password and Confirm password	The password the user will initially use when signing in.
Deny access	Select **Yes** to prevent this user from signing in to the tenant account, even though they might still belong to one or more groups. For example, select **Yes** to temporarily suspend a user's ability to sign in.

2. Select **Continue**.

Assign to groups

Steps

1. Assign the user to one or more local groups to determine which tasks they can perform.

   Assigning a user to groups is optional. If you'd prefer, you can select users when you create or edit groups.

   Users who don't belong to any groups will have no management permissions. Permissions are cumulative. Users will have all permissions for all groups they belong to. See Tenant management permissions.

2. Select **Create user**.

   If your tenant account has the **Use grid federation connection** permission and you are on the tenant's source grid, the new local user is cloned to the tenant's destination grid. **Success** appears as the **Cloning status** in the Overview section of the user's detail page.

3. Select **Finish** to return to the Users page.

View or edit local user

Steps

1. Select **ACCESS MANAGEMENT** > **Users**.
2. Review the information provided on the Users page, which lists basic information for all local and federated users for this tenant account.

   If the tenant account has the **Use grid federation connection** permission and you are viewing the user on the tenant's source grid:

   - A banner message indicates that if you edit or remove a user, your changes will not be synced to the other grid.

- As needed, a banner message indicates if users were not cloned to the tenant on the destination grid. You can retry a user clone that failed.

3. If you want to change the user's full name:
   a. Select the checkbox for the user.
   b. Select **Actions** > **Edit full name**.
   c. Enter the new name.
   d. Select **Save changes.**
4. If you want to view more details or make additional edits, do either of the following:
   - Select the username.
   - Select the checkbox for the user, and select **Actions** > **View user details**.
5. Review the Overview section, which shows the following information for each user:
   - Full name
   - Username
   - User type
   - Denied access
   - Access mode
   - Group membership
   - Additional fields if the tenant account has the **Use grid federation connection** permission and you are viewing the user on the tenant's source grid:
     - Cloning status, either **Success** or **Failure**
     - A blue banner indicating that if you edit this user, your changes will not be synced to the other grid.
6. Edit user settings as needed. See Create local user for details about what to enter.
   a. In the Overview section, change the full name by selecting the name or the edit icon .

      You can't change the username.

   b. On the **Password** tab, change the user's password, and select **Save changes**.
   c. On the **Access** tab, select **No** to allow the user to sign in or select **Yes** to prevent the user from signing in. Then, select **Save changes**.
   d. On the **Access keys** tab, select **Create key** and follow the instructions for creating another user's S3 access keys.
   e. On the **Groups** tab, select **Edit groups** to add the user to groups or remove the user from groups. Then, select **Save changes**.
7. Confirm that you selected **Save changes** for each section you changed.

Duplicate local user

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

> If your tenant account has the **Use grid federation connection** permission and you duplicate a user from the tenant's source grid, the duplicated user will be cloned to the tenant's destination grid.

Steps

1. Select **ACCESS MANAGEMENT** > **Users**.
2. Select the checkbox for the user you want to duplicate.
3. Select **Actions** > **Duplicate user**.
4. See Create local user for details about what to enter.
5. Select **Create user**.

Retry user clone

To retry a clone that failed:

1. Select each user that indicates *(Cloning failed)* below the user name.
2. Select **Actions** > **Clone users**.
3. View the status of the clone operation from the details page of each user you're cloning.

For additional information, see Clone tenant groups and users.

Delete one or more local users

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

> If your tenant account has the **Use grid federation connection** permission and you delete a local user, StorageGRID will not delete the corresponding user on the other grid. If you need to keep this information in sync, you must delete the same user from both grids.

> You must use the federated identity source to delete federated users.

Steps

1. Select **ACCESS MANAGEMENT** > **Users**.
2. Select the checkbox for each user you want to delete.
3. Select **Actions** > **Delete user** or **Actions** > **Delete users**.

   A confirmation dialog box appears.

4. Select **Delete user** or **Delete users**.

Manage S3 access keys

Manage 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.

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.

Create 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.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Manage your own S3 credentials or Root access 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, don't create more keys than you need, and delete the keys you aren't 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 don't need to periodically create new keys, you don't 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.

     > The expiration date can be a maximum of five years from the current date. The expiration time can be a minimum of one minute from the current time.

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.

   > Don't close this dialog box until you have copied or downloaded this information. You can't copy or download keys after the dialog box has been closed.

6. Select **Finish**.

   The new key is listed on the My access keys page.

7. If your tenant account has the **Use grid federation connection** permission, optionally use the Tenant Management API to manually clone S3 access keys from the tenant on the source grid to the tenant on the destination grid. See Clone S3 access keys using the API.

View 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.

> 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.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Manage your own S3 credentials permission.

Steps

1. Select **STORAGE (S3)** > **My access keys**.
2. From the My access keys page, sort any existing access keys by **Expiration time** or **Access key ID**.
3. As needed, create new keys or delete any 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.

Delete 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.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You 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**.
2. From the My access keys page, select the checkbox for each access key you want to remove.
3. Select **Delete key**.
4. From the confirmation dialog box, select **Delete key**.

   A confirmation message appears in the upper right corner of the page.

Create 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.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has 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, don't create more keys than the user needs, and delete the keys that aren't 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 don't need to periodically create new keys, you don't 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 **Don't 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.

   > The expiration date can be a maximum of five years from the current date. The expiration time can be a minimum of one minute from the current time.

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.

   > Don't close this dialog box until you have copied or downloaded this information. You can't copy or download keys after the dialog box has been closed.

7. Select **Finish**.

   The new key is listed on the Access keys tab of the user details page.

8. If your tenant account has the **Use grid federation connection** permission, optionally use the Tenant Management API to manually clone S3 access keys from the tenant on the source grid to the tenant on the destination grid. See Clone S3 access keys using the API.

View 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.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You 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**.

2. From the Users page, select the user whose S3 access keys you want to view.
3. From the User details page, select **Access keys**.
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

- Create another user's S3 access keys
- Delete another user's S3 access keys

Delete 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.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You 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**.
2. From the Users page, select the user whose S3 access keys you want to manage.
3. From the User details page, select **Access keys**, and then select the checkbox for each access key you want to delete.
4. Select **Actions** > **Delete selected key**.
5. From the confirmation dialog box, select **Delete key**.

   A confirmation message appears in the upper right corner of the page.

Manage S3 buckets

Create an S3 bucket

You can use the Tenant Manager to create S3 buckets for object data.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Root access or Manage all buckets permission. These permissions override the permissions settings in group or bucket policies.

> Permissions to set or modify S3 Object Lock properties of buckets or objects can be granted by bucket policy or group policy.

- If you plan to enable S3 Object Lock for a bucket, a grid admin has enabled the global S3 Object Lock setting for the StorageGRID system, and you have reviewed the requirements for S3 Object Lock buckets and objects.
- If each tenant will have 5,000 buckets, each Storage Node in the grid has a minimum of 64 GB of RAM.

> Each grid can have a maximum of 100,000 buckets.

Access the wizard

Steps

1. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.
2. Select **Create bucket**.

Enter details

Steps

1. Enter details for the bucket.

Field	Description
Bucket name	A name for the bucket that complies 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. • Each label must start and end with a lowercase letter or a number and can only use lowercase letters, numbers, and hyphens. • Must not contain periods in virtual hosted style requests. Periods will cause problems with server wildcard certificate verification. For more information, see the Amazon Web Services (AWS) documentation on bucket naming rules. **Note**: You can't change the bucket name after creating the bucket.

Field	Description
Region	The bucket's region. 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. **Note**: You can't change the region after creating the bucket.

2. Select **Continue**.

Manage settings

Steps

1. Optionally, enable object versioning for the bucket.

   Enable object versioning if you want to store every version of each object in this bucket. You can then retrieve previous versions of an object as needed. You must enable object versioning if the bucket will be used for cross-grid replication.

2. If the global S3 Object Lock setting is enabled, optionally enable S3 Object Lock for the bucket to store objects using a write-once-read-many (WORM) model.

   Enable S3 Object Lock for a bucket only if you need to keep objects for fixed amount of time, for example, to meet certain regulatory requirements. S3 Object Lock is a permanent setting that helps you prevent objects from being deleted or overwritten for a fixed amount of time or indefinitely.

   > After the S3 Object Lock setting is enabled for a bucket, it can't be disabled. Anyone with the correct permissions can add objects to this bucket that can't be changed. You might not be able to delete these objects or the bucket itself.

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

3. If you selected **Enable S3 Object Lock**, optionally enable **Default retention** for this bucket.

   > Your grid administrator must give you permission to use specific features of S3 Object Lock.

   When **Default retention** is enabled, new objects added to the bucket will be automatically protected from being deleted or overwritten. The **Default retention** setting does not apply to objects that have their own retention periods.

   a. If **Default retention** is enabled, specify a **Default retention mode** for the bucket.

Default retention mode	Description
Governance	• Users with the `s3:BypassGovernanceRetention` permission can use the `x-amz-bypass-governance-retention: true` request header to bypass retention settings. • These users can delete an object version before its retain-until-date is reached. • These users can increase, decrease, or remove an object's retain-until-date.
Compliance	• The object can't be deleted until its retain-until-date is reached. • The object's retain-until-date can be increased, but it can't be decreased. • The object's retain-until-date can't be removed until that date is reached. **Note**: Your grid administrator must allow you to use compliance mode.

b. If **Default retention** is enabled, specify the **Default retention period** for the bucket.

   The **Default retention period** indicates how long new objects added to this bucket should be retained, starting from the time they are ingested. Specify a value that is less than or equal to the maximum retention period for the tenant, as set by the grid administrator.

   A *maximum* retention period, which can be a value from 1 day to 100 years, is set when the grid administrator creates the tenant. When you set a *default* retention period, it can't exceed the value set for the maximum retention period. If needed, ask your grid administrator to increase or decrease the maximum retention period.

4. Optionally, select **Enable capacity limit**.

   Capacity limit is the maximum capacity available for this bucket's objects. This value represents a logical amount (object size), not a physical amount (size on disk).

   If no limit is set, the capacity for this bucket is unlimited. Refer to Capacity limit usage for more information.

5. Select **Create bucket**.

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

6. Optionally, select **Go to bucket details page** to view bucket details and perform additional configuration.

View bucket details

You can view the buckets in your tenant account.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.

- You belong to a user group that has the Root access, Manage all buckets, or View all buckets permission. These permissions override the permission settings in group or bucket policies.

Steps

1. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.

   The Buckets page appears.

2. Review the summary table for each bucket.

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

   > The Object Count, Space Used, and Usage values displayed are estimates. These estimates are affected by the timing of ingests, network connectivity, and node status. If buckets have versioning enabled, deleted object versions are included in the object count.

   Name
   The bucket's unique name, which can't be changed.

   Enabled features
   The list of features that are enabled for the bucket.

   S3 Object Lock
   Whether S3 Object Lock is enabled for the bucket.

   This column appears only if S3 Object Lock is enabled for the grid. This column also shows information for any legacy Compliant buckets.

   Region
   The bucket's region, which can't be changed. This column is hidden by default.

   Object count
   The number of objects in this bucket. If buckets have versioning enabled, non-current object versions are included in this value.

   When objects are added or deleted, this value might not update immediately.

   Space used
   The logical size of all objects in the bucket. The logical size does not include the actual space required for replicated or erasure-coded copies or for object metadata.

   This value can take up to 10 minutes to update.

   Usage
   The percentage used of the bucket's capacity limit, if one has been set.

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

This value indicates logical size, not physical size needed to store the objects and their metadata.

Capacity

If set, the capacity limit for the bucket.

Date created

The date and time the bucket was created. This column is hidden by default.

3. To view details for a specific bucket, select the bucket name from the table.
   a. View the summary information at the top of the web page to confirm the details for the bucket, such as Region and Object count.
   b. View the Capacity limit usage bar. If the usage is 100% or near 100%, consider increasing the limit or deleting some objects.
   c. As needed, select **Delete objects in bucket** and **Delete bucket**.

Pay close attention to the cautions that appear when you select each of these options. For more information, refer to:

- Delete all objects in a bucket
- Delete a bucket (bucket must be empty)

   d. View or change settings for the bucket in each of the tabs as needed.
      - **S3 Console**: View the objects for the bucket. For more information, refer to Use S3 Console.
      - **Bucket options**: View or change option settings. Some settings, such as S3 Object Lock, can't be changed after the bucket is created.
        - Manage bucket consistency
        - Last access time updates
        - Capacity limit
        - Object versioning
        - S3 Object Lock
        - Default bucket retention
        - Manage cross-grid replication (if allowed for the tenant)
      - **Platform services**: Manage platform services (if allowed for the tenant)
      - **Bucket access**: View or change option settings. You must have specific access permissions.
        - Configure Cross-Origin Resource Sharing (CORS) so the bucket and objects in the bucket will be accessible to web applications in other domains.
        - Control user access for an S3 bucket and objects in that bucket.

Apply an ILM policy tag to a bucket

Choose an ILM policy tag to apply to a bucket based on your object storage requirements.

The ILM policy controls where the object data is stored and whether it is deleted after a certain time period. Your grid administrator creates ILM policies and assigns them to ILM policy tags when using multiple active

policies.

Avoid frequently reassigning a bucket's policy tag. Otherwise, performance issues might occur.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Root access, Manage all buckets, or View all buckets permission. These permissions override the permission settings in group or bucket policies.

Steps

1. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.

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

2. Select the name of the bucket you want to assign an ILM policy tag to.

   You can also change the ILM policy tag assignment for a bucket that already has a tag assigned.

   The Object Count and Space Used values displayed are estimates. These estimates are affected by the timing of ingests, network connectivity, and node status. If buckets have versioning enabled, deleted object versions are included in the object count.

3. In the Bucket options tab, expand the ILM policy tag accordion. This accordion only appears if your grid administrator has enabled the use of custom policy tags.
4. Read the description of each policy tag to determine which tag should be applied to the bucket.

   Changing the ILM policy tag for a bucket will trigger ILM reevaluation of all objects in the bucket. If the new policy retains objects for a limited time, older objects will be deleted.

5. Select the radio button for the tag you want to assign to the bucket.
6. Select **Save changes**. A new S3 bucket tag will be set on the bucket with the key `NTAP-SG-ILM-BUCKET-TAG` and the value of the ILM policy tag name.

   Ensure that your S3 applications do not accidentally override or delete the new bucket tag. If this tag is omitted when applying a new TagSet to the bucket, objects in the bucket will revert to being evaluated against the default ILM policy.

   Set and modify ILM policy tags using only the Tenant Manager or Tenant Manager API where the ILM policy tag is validated. Do not modify the `NTAP-SG-ILM-BUCKET-TAG` ILM policy tag using the S3 PutBucketTagging API or the S3 DeleteBucketTagging API.

   Changing the policy tag assigned to a bucket has a temporary performance impact while objects are being reevaluated using the new ILM policy.

Manage bucket policy

You can control user access for an S3 bucket and the objects in that bucket.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Root access permission. The View all buckets and Manage all buckets permissions only allow viewing.
- You've verified that the required number of Storage Nodes and sites are available. If two or more Storage Nodes are not available within any site, or if a site is not available, changes to these settings might not be available.

Steps

1. Select **Buckets**, then select the bucket you want to manage.
2. On the bucket details page, select **Bucket access** > **Bucket policy**.
3. Do one of the following:
   - Enter a bucket policy by selecting the **Enable policy** checkbox. Then enter a valid JSON formatted string.

     Each bucket policy has a size limit of 20,480 bytes.
   - Modify an existing policy by editing the string.
   - Disable a policy by unselecting **Enable policy**.

For detailed information about bucket policies, including language syntax and examples, see Example bucket policies.

Manage bucket consistency

Consistency values can be used to specify the availability of bucket setting changes as well as to provide a balance between the availability of the objects within a bucket and the consistency of those objects across different Storage Nodes and sites. You can change the consistency values to be different from the default values so that client applications can meet their operational needs.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Manage all buckets or Root access permission. These permissions override the permissions settings in group or bucket policies.

Bucket consistency guidelines

The bucket consistency is used to determine the consistency for client applications affecting objects within that S3 bucket. In general, you should use the **Read-after-new-write** consistency for your buckets.

Change bucket consistency

If the **Read-after-new-write** consistency does not meet the client application's requirements, you can change the consistency by setting the bucket consistency or by using the `Consistency-Control` header. The `Consistency-Control` header overrides the bucket consistency.

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

Steps

1. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.
2. Select the bucket name from the table.

   The bucket details page appears.

3. From the **Bucket options** tab, select the ** accordion.
4. Select a consistency for operations performed on the objects in this bucket.
   - **All**: Provides the highest level 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. Recommended for most cases.
   - **Available**: Provides eventual consistency for both new objects and object updates. For S3 buckets, use only as required (for example, for a bucket that contains log values that are rarely read, or for HEAD or GET operations on keys that don't exist). Not supported for S3 FabricPool buckets.
5. Select **Save changes**.

What happens when you change bucket settings

Buckets have multiple settings that affect the behavior of the buckets and the objects within those buckets.

The following bucket settings use **strong** consistency by default. If two or more Storage Nodes are not available within any site, or if a site is not available, any changes to these settings might not be available.

- Background empty bucket deletion
- Last Access Time
- Bucket lifecycle
- Bucket policy
- Bucket tagging
- Bucket versioning
- S3 Object Lock
- Bucket encryption

> The consistency value for bucket versioning, S3 Object Lock, and bucket encryption cannot be set to a value that is not strongly consistent.

The following bucket settings do not use strong consistency and have higher availability for changes. Changes to these settings might take some time before having an effect.

- Platform services configuration: Notification, Replication, or Search integration
- CORS configuration
- Change bucket consistency

If the default consistency used when changing bucket settings does not meet the client application's requirements, you can change the consistency by using the `Consistency-Control` header for the S3 REST API or by using the `reducedConsistency` or `force` options in the Tenant Management API.

Enable or disable 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 as an advanced filter or as a reference time. You can ignore these instructions if your StorageGRID system does not include such a rule. See Use Last access time in ILM rules for details.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Manage all buckets or Root access permission. These permissions override the permissions settings in group or bucket policies.

About this task

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.

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 list objects or object versions	No	No	No	No
Request to copy an object from one bucket to another	• No, for the source copy • Yes, for the destination copy	• No, for the source copy • Yes, for the destination copy	• Yes, for the source copy • Yes, for the destination copy	• 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 **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.
2. Select the bucket name from the table.

   The bucket details page appears.

3. From the **Bucket options** tab, select the **Last access time updates** accordion.
4. Enable or disable last access time updates.
5. Select **Save changes**.

Change object versioning for a bucket

If you are using an S3 tenant, you can change the versioning state for S3 buckets.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Manage all buckets or Root access permission. These permissions override the permissions settings in group or bucket policies.
- You've verified that the required number of Storage Nodes and sites are available. If two or more Storage Nodes are not available within any site, or if a site is not available, changes to these settings might not be available.

About this task

You can enable or suspend object versioning for a bucket. After you enable versioning for a bucket, it can't return to an unversioned state. However, you can suspend versioning for the bucket.

- Disabled: Versioning has never been enabled
- Enabled: Versioning is enabled
- Suspended: Versioning was previously enabled and is suspended

For more information, see the following:

- Object versioning
- ILM rules and policies for S3 versioned objects (Example 4)
- How objects are deleted

Steps

1. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.
2. Select the bucket name from the table.

   The bucket details page appears.

3. From the **Bucket options** tab, select the **Object versioning** accordion.
4. Select a versioning state for the objects in this bucket.

   Object versioning must remain enabled for a bucket used for cross-grid replication. If S3 Object Lock or legacy compliance is enabled, the **Object versioning** options are disabled.

Option	Description
Enable versioning	Enable object versioning if you want to store every version of each object in this bucket. You can then retrieve previous versions of an object as needed. Objects that were already in the bucket will be versioned when they are modified by a user.
Suspend versioning	Suspend object versioning if you no longer want new object versions to be created. You can still retrieve any existing object versions.

5. Select **Save changes**.

Use S3 Object Lock to retain objects

You can use S3 Object Lock if buckets and objects must comply with regulatory requirements for retention.

Your grid administrator must give you permission to use specific features of S3 Object Lock.

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).

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, bucket versioning is required and is enabled automatically.

A bucket without S3 Object Lock can only have objects without retention settings specified. No ingested objects will have retention settings.

A bucket with S3 Object Lock can have objects with and without retention settings specified by S3 client applications. Some objects ingested will have retention settings.

A bucket with S3 Object Lock and default retention configured can have uploaded objects with retention settings specified and new objects without retention settings. The new objects use the default setting, because the retention setting hasn't been configured at the object-level.

Effectively, all newly ingested objects have retention settings when default retention is configured. Existing objects without object retention settings remain unaffected.

Retention modes

The StorageGRID S3 Object Lock feature supports two retention modes to apply different levels of protection to objects. These modes are equivalent to the Amazon S3 retention modes.

- In compliance mode:
  - The object can't be deleted until its retain-until-date is reached.
  - The object's retain-until-date can be increased, but it can't be decreased.
  - The object's retain-until-date can't be removed until that date is reached.
- In governance mode:
  - Users with special permission can use a bypass header in requests to modify certain retention settings.
  - These users can delete an object version before its retain-until-date is reached.
  - These users can increase, decrease, or remove an object's retain-until-date.

Retention settings for object versions

If a bucket is created with S3 Object Lock enabled, users can use the S3 client application to optionally specify the following retention settings for each object that is added to the bucket:

- **Retention mode**: Either compliance or governance.
- **Retain-until-date**: If an object version's retain-until-date is in the future, the object can be retrieved, but it can't be deleted.
- **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.

  If an object is under a legal hold, no one can delete the object, regardless of its retention mode.

For details on the object settings, see Use S3 REST API to configure S3 Object Lock.

Default retention setting for buckets

If a bucket is created with S3 Object Lock enabled, users can optionally specify the following default settings for the bucket:

- **Default retention mode**: Either compliance or governance.
- **Default retention period**: How long new object versions added to this bucket should be retained, starting from the day they are added.

The default bucket settings apply only to new objects that don't have their own retention settings. Existing bucket objects aren't affected when you add or change these default settings.

See Create an S3 bucket and Update S3 Object Lock default retention.

S3 Object Lock tasks

The following lists for grid administrators and tenant users contain the high-level tasks for using the S3 Object Lock feature.

Grid administrator

- Enable global S3 Object Lock setting for entire StorageGRID system.
- Ensure that information lifecycle management (ILM) policies are *compliant*; that is, they meet the requirements of buckets with S3 Object Lock enabled.
- As needed, allow a tenant to use Compliance as the retention mode. Otherwise, only Governance mode is allowed.
- As needed, set a maximum retention period for a tenant.

Tenant user

- Review considerations for buckets and objects with S3 Object Lock.
- As needed, contact grid administrator to enable global S3 Object Lock setting and set permissions.
- Create buckets with S3 Object Lock enabled.
- Optionally, configure default retention settings for a bucket:
  - Default retention mode: Governance or Compliance, if allowed by the grid administrator.
  - Default retention period: Must be less than or equal to maximum retention period set by grid administrator.
- Use the S3 client application to add objects and optionally set object-specific retention:
  - Retention mode. Governance or Compliance, if allowed by the grid administrator.
  - Retain Until Date: Must be less than or equal to what is allowed by the maximum retention period set by grid administrator.

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.
- If you plan to use S3 Object Lock, you must enable S3 Object Lock when you create the bucket. You can't enable S3 Object Lock for an existing bucket.
- When S3 Object Lock is enabled for a bucket, StorageGRID automatically enables versioning for that

bucket. You can't disable S3 Object Lock or suspend versioning for the bucket.

- Optionally, you can specify a default retention mode and retention period for each bucket using the Tenant Manager, the Tenant Management API, or the S3 REST API. The bucket's default retention settings apply only to new objects added to the bucket that don't have their own retention settings. You can override these default settings by specifying a retention mode and retain-until-date for each object version when it is uploaded.
- Bucket lifecycle configuration is supported for buckets with S3 Object Lock enabled.
- CloudMirror replication is not supported for buckets with S3 Object Lock enabled.

Requirements for objects in buckets with S3 Object Lock enabled

- To protect an object version, you can specify default retention settings for the bucket, or you can specify retention settings for each object version. Object-level retention settings can be specified using the S3 client application or the S3 REST API.
- 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 these stages:

1. **Object ingest**

   When an object version is added to bucket that has S3 Object Lock enabled, retention settings are applied as follows:

   - If retention settings are specified for the object, the object-level settings are applied. Any default bucket settings are ignored.
   - If no retention settings are specified for the object, the default bucket settings are applied, if they exist.
   - If no retention settings are specified for the object or the bucket, the object is not protected by S3 Object Lock.

   If retention settings are applied, both the object and any S3 user-defined metadata are protected.

2. **Object retention and deletion**

   Multiple copies of each protected object are stored by StorageGRID for the specified retention period. The exact number and type of object copies and the storage locations are determined by the compliant rules in the active ILM policies. Whether a protected object can be deleted before its retain-until-date is reached depends on its retention mode.

   - If an object is under a legal hold, no one can delete the object, regardless of its retention mode.

Can I still manage 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
NetApp Knowledge Base: How to manage legacy Compliant buckets in StorageGRID 11.5.

Update S3 Object Lock default retention

If you enabled S3 Object Lock when you created the bucket, you can edit the bucket to change the default retention settings. You can enable (or disable) default retention and set a default retention mode and retention period.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Manage all buckets or Root access permission. These permissions override the permissions settings in group or bucket policies.
- S3 Object Lock is enabled globally for your StorageGRID system, and you enabled S3 Object Lock when you created the bucket. See Use S3 Object Lock to retain objects.

Steps

1. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.
2. Select the bucket name from the table.

   The bucket details page appears.

3. From the **Bucket options** tab, select the **S3 Object Lock** accordion.
4. Optionally, enable or disable **Default retention** for this bucket.

   Changes to this setting don't apply to objects already in the bucket or to any objects that might have their own retention periods.

5. If **Default retention** is enabled, specify a **Default retention mode** for the bucket.

Default retention mode	Description
Governance	• Users with the `s3:BypassGovernanceRetention` permission can use the `x-amz-bypass-governance-retention: true` request header to bypass retention settings. • These users can delete an object version before its retain-until-date is reached. • These users can increase, decrease, or remove an object's retain-until-date.
Compliance	• The object can't be deleted until its retain-until-date is reached. • The object's retain-until-date can be increased, but it can't be decreased. • The object's retain-until-date can't be removed until that date is reached. **Note**: Your grid administrator must allow you to use compliance mode.

6. If **Default retention** is enabled, specify the **Default retention period** for the bucket.

   The **Default retention period** indicates how long new objects added to this bucket should be retained,

starting from the time they are ingested. Specify a value that is less than or equal to the maximum retention period for the tenant, as set by the grid administrator.

A *maximum* retention period, which can be a value from 1 day to 100 years, is set when the grid administrator creates the tenant. When you set a *default* retention period, it can't exceed the value set for the maximum retention period. If needed, ask your grid administrator to increase or decrease the maximum retention period.

7. Select **Save changes**.

Configure 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.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- For GET CORS configuration requests, you belong to a user group that has the Manage all buckets or View all buckets permission. These permissions override the permissions settings in group or bucket policies.
- For PUT CORS configuration requests, you belong to a user group that has the Manage all buckets permission. This permission overrides the permissions settings in group or bucket policies.
- The Root access permission provides access to all CORS configuration requests.

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`.

Enable CORS for a bucket

Steps

1. Use a text editor to create the required XML. This example shows the XML used to enable CORS for an S3 bucket. Specifically:
   - Allows any domain to send GET requests to the bucket
   - Only allows the `http://www.example.com` domain to send GET, 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 User Guide.

2. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.
3. Select the bucket name from the table.

   The bucket details page appears.

4. From the **Bucket access** tab, select the **Cross-Origin Resource Sharing (CORS)** accordion.
5. Select the **Enable CORS** checkbox.
6. Paste the CORS configuration XML into the text box.
7. Select **Save changes**.

Modify CORS setting

Steps

1. Update the CORS configuration XML in the text box, or select **Clear** to start over.
2. Select **Save changes**.

Disable CORS setting

Steps

1. Clear the **Enable CORS** checkbox.
2. Select **Save changes**.

Delete objects in bucket

You can use the Tenant Manager to delete the objects in one or more buckets.

Considerations and requirements

Before performing these steps, note the following:

- When you delete the objects in a bucket, StorageGRID permanently removes all objects and all object versions in each selected bucket from all nodes and sites in your StorageGRID system. StorageGRID also removes any related object metadata. You will not be able to recover this information.
- Deleting all of the objects in a bucket might take minutes, days, or even weeks, based on the number of objects, object copies, and concurrent operations.
- If a bucket has S3 Object Lock enabled, it might remain in the **Deleting objects: read-only** state for *years*.

> A bucket that uses S3 Object Lock will remain in the **Deleting objects: read-only** state until the retention date is reached for all objects and any legal holds are removed.

- While objects are being deleted, the bucket's state is **Deleting objects: read-only**. In this state, you can't add new objects to the bucket.
- When all objects have been deleted, the bucket remains in the read-only state. You can do one of the following:
  - Return the bucket to write mode and reuse it for new objects
  - Delete the bucket
  - Keep the bucket in read-only mode to reserve its name for future use
- If a bucket has object versioning enabled, delete markers that were created in StorageGRID 11.8 or later can be removed using the Delete objects in bucket operations.
- If a bucket has object versioning enabled, the delete objects operation will not remove delete markers that were created in StorageGRID 11.7 or earlier. See information about deleting objects in a bucket in How S3 versioned objects are deleted.
- If you use cross-grid replication, note the following:
  - Using this option does not delete any objects from the bucket on the other grid.
  - If you select this option for the source bucket, the **Cross-grid replication failure** alert will be triggered if you add objects to the destination bucket on the other grid. If you can't guarantee no one will add objects to the bucket on the other grid, disable cross-grid replication for that bucket before deleting all bucket objects.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Root access permission. This permission overrides the permissions settings in group or bucket policies.

Steps

1. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.

   The Buckets page appears and shows all existing S3 buckets.

2. Use the **Actions** menu or the details page for a specific bucket.

Actions menu

a. Select the checkbox for each bucket you want to delete objects from.

b. Select **Actions** > **Delete objects in bucket**.

Details page

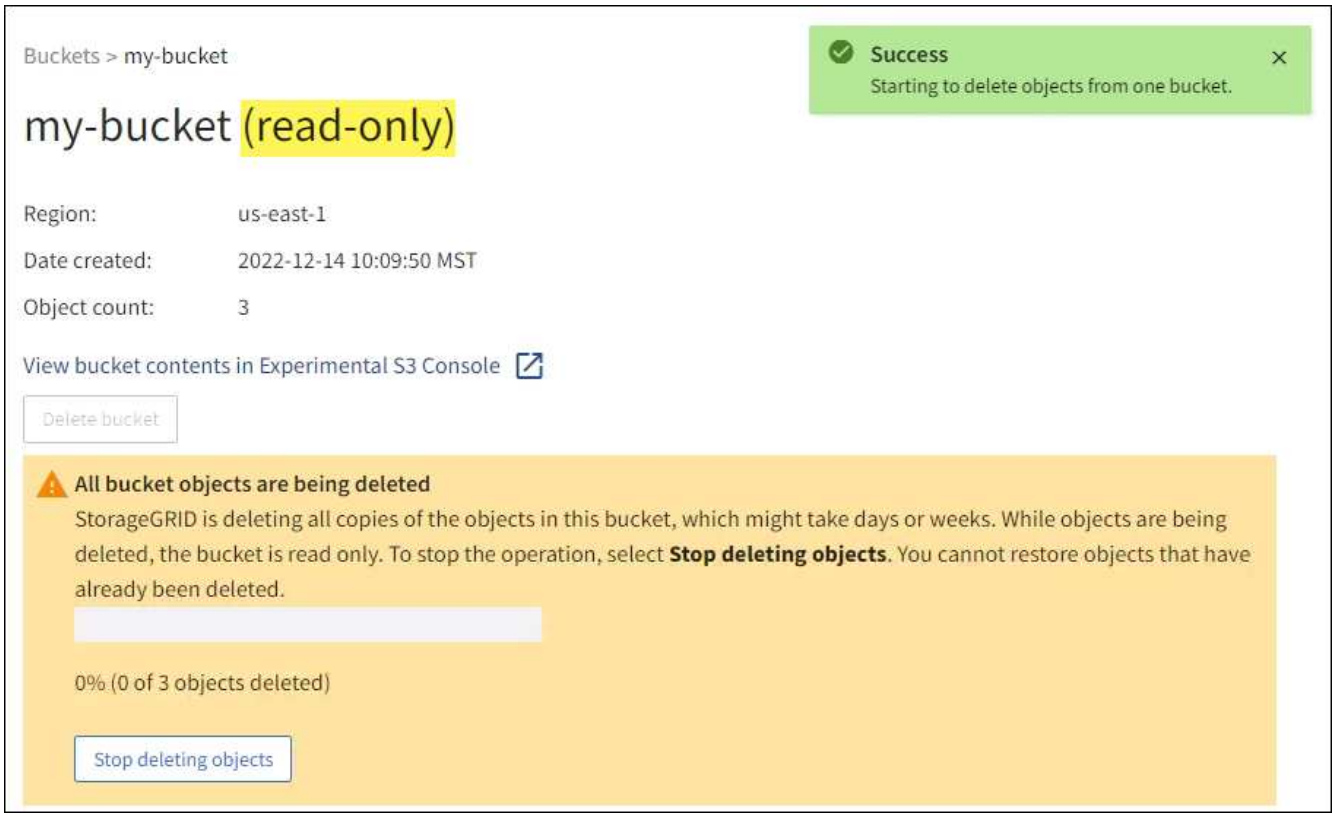
a. Select a bucket name to display its details.

b. Select **Delete objects in bucket**.

3. When the confirmation dialog box appears, review the details, enter **Yes**, and select **OK**.
4. Wait for the delete operation to begin.

   After a few minutes:

   - A yellow status banner appears on the bucket details page. The progress bar represents what percentage of objects have been deleted.
   - **(read-only)** appears after the bucket's name on the bucket details page.
   - **(Deleting objects: read-only)** appears next to the bucket's name on the Buckets page.

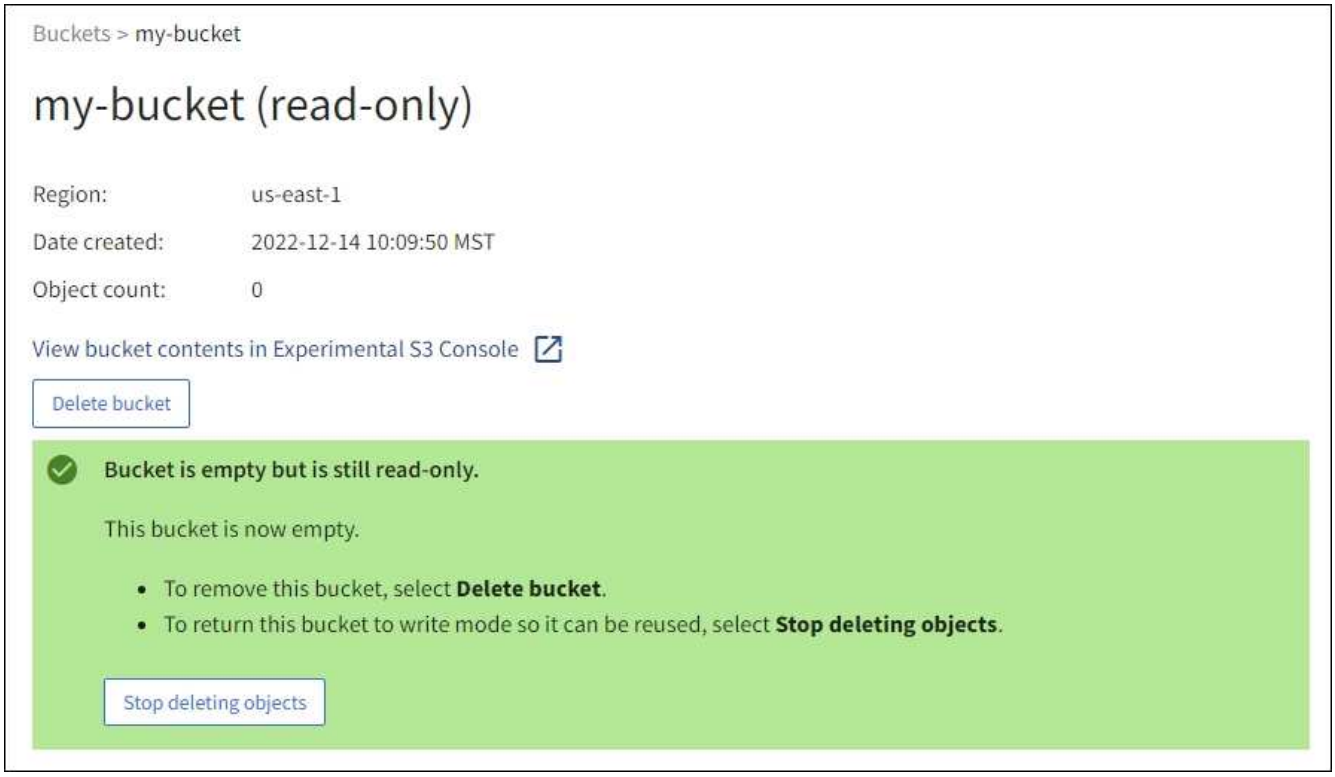


5. As required while the operation is running, select **Stop deleting objects** to halt the process. Then, optionally, select **Delete objects in bucket** to resume the process.

   When you select **Stop deleting objects**, the bucket is returned to write mode; however, you can't access or restore any objects that have been deleted.

6. Wait for the operation to complete.

When the bucket is empty, the status banner is updated, but the bucket remains read only.



7. Do one of the following:
   - Exit the page to keep the bucket in read-only mode. For example, you might keep an empty bucket in read-only mode to reserve the bucket name for future use.
   - Delete the bucket. You can select **Delete bucket** to delete a single bucket or return the Buckets page and select **Actions** > **Delete** buckets to remove more than one bucket.

     If you are unable to delete a versioned bucket after all objects were deleted, delete markers might remain. To delete the bucket, you must remove all remaining delete markers.

   - Return the bucket to write mode and optionally reuse it for new objects. You can select **Stop deleting objects** for a single bucket or return to the Buckets page and select **Action** > **Stop deleting objects** for more than one bucket.

Delete S3 bucket

You can use the Tenant Manager to delete one or more S3 buckets that are empty.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Manage all buckets or Root access permission. These permissions override the permissions settings in group or bucket policies.
- The buckets you want to delete are empty. If buckets you want to delete are *not* empty, delete objects from the bucket.

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 can't delete an S3 bucket if it contains objects, noncurrent object versions, or delete markers. For information about how S3 versioned objects are deleted, see How objects are deleted.

Steps

1. Select **View buckets** from the dashboard, or select **STORAGE (S3)** > **Buckets**.

   The Buckets page appears and shows all existing S3 buckets.

2. Use the **Actions** menu or the details page for a specific bucket.

   Actions menu

   a. Select the checkbox for each bucket you want to delete.

   b. Select **Actions** > **Delete buckets**.

   Details page

   a. Select a bucket name to display its details.

   b. Select **Delete bucket**.

3. When the confirmation dialog box appears, select **Yes**.

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

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

Use S3 Console

You can use S3 Console to view and manage the objects in an S3 bucket.

S3 Console allows you to:

- Upload, download, rename, copy, move, and delete objects
- View, revert, download, and delete object versions
- Search for objects by prefix
- Manage object tags
- View object metadata
- View, create, rename, copy, move, and delete folders

S3 Console provides an improved user experience for the most common cases. It is not designed to replace CLI or API operations in all situations.

If using S3 Console results in operations taking too long (for example, minutes or hours), consider:

- Reducing the number of selected objects
- Using non-graphical (API or CLI) methods to access your data

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- If you want to manage objects, you belong to a user group that has the Root access permission. Alternatively, you belong to a user group that has the Use S3 Console tab permission and either the View all buckets permission or Manage all buckets permission. See Tenant management permissions.
- An S3 Group or Bucket policy has been configured for the user. See Use bucket and group access policies.
- You know the user's access key ID and secret access key. Optionally, you have a `.csv` file containing this information. See the instructions for creating access keys.

Steps

1. Select **STORAGE** > **Buckets** > ***bucket name***.
2. Select the S3 Console tab.
3. Paste the access key ID and secret access key into the fields. Otherwise, select **Upload access keys** and select your `.csv` file.
4. Select **Sign in**.
5. The table of bucket objects appears. You can manage objects as needed.

Additional information

- **Search by prefix**: The prefix search feature only searches for objects that begin with a specific word relative to the current folder. The search does not include objects that contain the word elsewhere. This rule also applies to objects within folders. For example, a search for `folder1/folder2/somefile-` would return objects that are within the `folder1/folder2/` folder and begin with the word `somefile-`.
- **Drag and drop**: You can drag and drop files from your computer's file manager to S3 Console. However, you cannot upload folders.
- **Operations on folders**: When you move, copy, or rename a folder, all objects in the folder are updated one at a time, which might take time.
- **Permanent deletion when bucket versioning is disabled**: When you overwrite or delete an object in a bucket with versioning disabled, the operation is permanent. See Change object versioning for a bucket.

Manage S3 platform services

S3 platform services

Platform services overview and considerations

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

For information about S3, see Use S3 REST API.

Overview of platform services

StorageGRID platform services can help you implement a hybrid cloud strategy by allowing you to send event notifications and copies of S3 objects and object metadata to external destinations.

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 (S3), 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, an Amazon SNS topic, or an Elasticsearch cluster hosted locally, on AWS, or elsewhere.

After you create an external 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:

- 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.
- If you also want to send notifications when these objects are stored to the bucket, you must add a notifications configuration.
- 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	Refer to
CloudMirror replication	• GetBucketReplication • PutBucketReplication	• CloudMirror replication • Operations on buckets

Platform service	S3 REST API	Refer to
Notifications	• GetBucketNotificationConfiguration • PutBucketNotificationConfiguration	• Notifications • Operations on buckets
Search integration	• GET Bucket metadata notification configuration • PUT Bucket metadata notification configuration	• Search integration • StorageGRID custom operations

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.
Destination endpoint throttling	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. 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. 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.
Ordering guarantees	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. 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.

Consideration	Details
ILM-driven object deletions	To match the deletion behavior of the AWS CRR and Amazon Simple Notification Service, CloudMirror and event notification requests aren't 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. In contrast, search integration requests are sent when objects are deleted because of ILM.
Using Kafka endpoints	For Kafka endpoints, Mutual TLS is not supported. As a result, if you have `ssl.client.auth` set to `required` in your Kafka broker configuration, it might cause Kafka endpoint configuration issues. The authentication of Kafka endpoints uses the following authentication types. These types are different from those used for the authentication of other endpoints, such as Amazon SNS, and require username and password credentials. • SASL/PLAIN • SASL/SCRAM-SHA-256 • SASL/SCRAM-SHA-512 **Note:** Configured storage proxy settings do not apply to Kafka platform services endpoints.

Considerations for using CloudMirror replication service

Consideration	Details
Replication status	StorageGRID does not support the `x-amz-replication-status` header.
Object size	The maximum size for objects that can be replicated to a destination bucket by the CloudMirror replication service is 5 TiB, which is the same as the maximum *supported* object size. **Note**: The maximum *recommended* size for a single PutObject operation is 5 GiB (5,368,709,120 bytes). If you have objects that are larger than 5 GiB, use multipart upload instead.
Bucket versioning and version IDs	If the source S3 bucket in StorageGRID has versioning enabled, you should also enable versioning for the destination bucket. 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. **Note**: Version IDs for the source bucket in StorageGRID aren't related to the version IDs for the destination bucket.

Consideration	Details
Tagging for object versions	The CloudMirror service does not replicate any PutObjectTagging or DeleteObjectTagging requests that supply a version ID, due to limitations in the S3 protocol. Because version IDs for the source and destination aren't related, there is no way to ensure that a tag update to a specific version ID will be replicated. In contrast, the CloudMirror service does replicate PutObjectTagging requests or DeleteObjectTagging requests that don't 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.
Multipart uploads and `ETag` values	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.
Objects encrypted with SSE-C (server-side encryption with customer-provided keys)	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.
Bucket with S3 Object Lock enabled	Replication is not supported for source or destination buckets with S3 Object Lock enabled.

Understand 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 external destination buckets.

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.

CloudMirror and ILM

CloudMirror replication operates independently of the grid's active ILM policies. 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.

CloudMirror and cross-grid replication

CloudMirror replication has important similarities and differences with the cross-grid replication feature. Refer to Compare cross-grid replication and CloudMirror replication.

CloudMirror and S3 buckets

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.

Existing buckets

When you enable CloudMirror replication for an existing bucket, only the new objects added to that bucket are replicated. Any existing objects in the bucket aren't 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 Amazon 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.

Multiple destination buckets

To replicate objects in a single bucket to multiple destination buckets, specify the destination for each rule in the replication configuration XML. You can't replicate an object to more than one bucket at the same time.

Versioned or unversioned buckets

You can configure CloudMirror replication on versioned or unversioned buckets. The destination buckets can be versioned or unversioned. 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, replication loops, and events

Deletion behavior

Is the same as the deletion behavior of the Amazon S3 service, Cross-Region Replication (CRR). 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 doesn't replicate the delete marker to the destination bucket or delete the destination object.

Protection from replication loops

As objects are replicated to the destination bucket, StorageGRID marks them as "replicas." A destination StorageGRID bucket won't replicate objects marked as replicas again, protecting you from accidental replication loops. This replica marking is internal to StorageGRID and doesn't 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 bidirectional CloudMirror between two grids.

Events in the destination bucket

The uniqueness and ordering of events in the destination bucket aren't 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.

Understand 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 Kafka cluster or Amazon Simple Notification Service.

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.

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 aren't 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.

StorageGRID event notifications follow the Amazon S3 API with some limitations.

- The following event types are supported:
  - s3:ObjectCreated:
  - s3:ObjectCreated:Put
  - s3:ObjectCreated:Post
  - s3:ObjectCreated:Copy
  - s3:ObjectCreated:CompleteMultipartUpload
  - s3:ObjectRemoved:
  - s3:ObjectRemoved:Delete
  - s3:ObjectRemoved:DeleteMarkerCreated
  - s3:ObjectRestore:Post
- Event notifications sent from StorageGRID use the standard JSON format but don't 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`

Understand 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 is created or deleted, or its metadata or tags are 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.

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 metadata sent to Elasticsearch.

> 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 from the "notification configuration XML" used to enable *event* notifications.

Search integration and S3 buckets

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.

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 can't edit the document's field types in the index.

Search notifications

Metadata 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 aren't 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.

Search integration service and Elasticsearch

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 NetApp Interoperability Matrix Tool to determine the supported versions of Elasticsearch.

Manage platform services endpoints

Configure 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. For a single tenant, you can configure a maximum of 500 platform services endpoints. Contact your StorageGRID administrator for more information.

What is a platform services endpoint?

A platform services endpoint specifies the information that StorageGRID needs to access the external destination.

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

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 Amazon Simple Notification Service (Amazon SNS) topic and notifications about object deletion to a second Amazon 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 Amazon SNS and Kafka endpoints. Simple Queue Service (SQS) or AWS Lambda endpoints aren't supported.

For Kafka endpoints, Mutual TLS is not supported. As a result, if you have `ssl.client.auth` set to `required` in your Kafka broker configuration, it might cause Kafka endpoint configuration issues.

Endpoints for the search integration service

StorageGRID supports search integration endpoints that represent Elasticsearch clusters. These Elasticsearch clusters can be in a local data center 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 don't 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

Specify URN for 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 a 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:mysite`, as follows:

- If the service is hosted on Amazon Web Services (AWS), use `arn:aws`
- If the service is hosted on Google Cloud Platform (GCP), use `arn:aws`
- If the service is hosted locally, use `urn:mysite`

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` or `kafka`
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` or `kafka-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 and GCP

For AWS and GCP 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:

  Specify an Amazon Simple Notification Service endpoint:

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

  Specify a Kafka endpoint:

  ```
  urn:mysite:kafka:optional:optional:kafka-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.

Create platform services endpoint

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

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- Platform services were enabled for your tenant account by a StorageGRID administrator.
- You belong to a user group that has the Manage endpoints or Root access permission.
- The resource referenced by the platform services endpoint have been created:
  - CloudMirror replication: S3 bucket
  - Event notification: Amazon Simple Notification Service (Amazon SNS) or Kafka topic
  - Search notification: Elasticsearch index, if the destination cluster is not configured to automatically create indexes.
- You 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)

Specify URN for platform services endpoint

- Authentication credentials (if required):

  Search integration endpoints

  For search integration endpoints, you can use the following credentials:

  - Access Key: Access key ID and secret access key
  - Basic HTTP: Username and password

  CloudMirror replication endpoints

  For CloudMirror replication endpoints, you can use the following credentials:

  - Access Key: Access key ID and secret access key
  - CAP (C2S Access Portal): Temporary credentials URL, server and client certificates, client keys, and an optional client private key passphrase.

  Amazon SNS endpoints

  For Amazon SNS endpoints, you can use the following credentials:

  - Access Key: Access key ID and secret access key

  Kafka endpoints

  For Kafka endpoints, you can use the following credentials:

  - SASL/PLAIN: Username and password
  - SASL/SCRAM-SHA-256: Username and password
  - SASL/SCRAM-SHA-512: Username and password

- Security certificate (if using a custom CA certificate)

- If the Elasticsearch security features are enabled, you have the monitor cluster privilege for connectivity testing, and either the write index privilege or both the index and delete index privileges for document updates.

Steps

1. Select **STORAGE (S3)** > **Platform services endpoints**. The Platform services endpoints page appears.
2. Select **Create endpoint**.
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 don't 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 don't specify a port, the following default ports are used:

- Port 443 for HTTPS URIs and port 80 for HTTP URIs (most endpoints)
- Port 9092 for HTTPS and HTTP URIs (Kafka endpoints only)

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 an 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, don't 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 can't change an endpoint's URN after the endpoint has been created.

6. Select **Continue**.
7. Select a value for **Authentication type**.

Search integration endpoints

Enter or upload the credentials for a search integration endpoint.

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.	• Access key ID • Secret access key
Basic HTTP	Uses a username and password to authenticate connections to the destination.	• Username • Password

CloudMirror replication endpoints

Enter or upload the credentials for a CloudMirror replication endpoint.

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.	• Access key ID • Secret access key
CAP (C2S Access Portal)	Uses certificates and keys to authenticate connections to the destination.	• Temporary credentials URL • Server CA certificate (PEM file upload) • Client certificate (PEM file upload) • Client private key (PEM file upload, OpenSSL encrypted format or unencrypted private key format) • Client private key passphrase (optional)

Amazon SNS endpoints

Enter or upload the credentials for an Amazon SNS endpoint.

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.	• Access key ID • Secret access key

Kafka endpoints

Enter or upload the credentials for a Kafka endpoint.

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.
SASL/PLAIN	Uses a username and password with plain text to authenticate connections to the destination.	• Username • Password
SASL/SCRAM-SHA-256	Uses a username and password using a challenge-response protocol and SHA-256 hashing to authenticate connections to the destination.	• Username • Password
SASL/SCRAM-SHA-512	Uses a username and password using a challenge-response protocol and SHA-512 hashing to authenticate connections to the destination.	• Username • Password

Select **Use delegation taken authentication** if the username and password are derived from a delegation token that was obtained from a Kafka cluster.

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

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 Grid 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 aren't 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

- Specify URN for platform services endpoint
- Configure CloudMirror replication
- Configure event notifications
- Configure search integration service

Test connection for 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.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Manage endpoints or Root access 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.

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

   The endpoint details page appears.

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**.

Edit 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 can't change the URN for a platform services endpoint.

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Manage endpoints or Root access 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.

2. Select the endpoint you want to edit.

   The endpoint details page appears.

3. Select **Configuration**.
4. As needed, change the configuration of the endpoint.

   > ⓘ You can't 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 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**.
      - For CAP (C2S Access Portal) authentication, change the temporary credentials URL or optional client private key passphrase and upload new certificate and key files as needed.

        > ⓘ The Client private key must be in OpenSSL encrypted format or unencrypted private key format.

   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**.

Delete platform services endpoint

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

Before you begin

- You are signed in to the Tenant Manager using a supported web browser.
- You belong to a user group that has the Manage endpoints or Root access 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.

2. Select the checkbox 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.

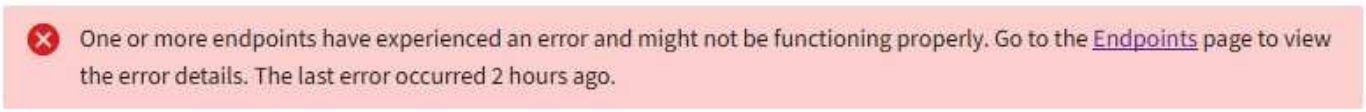
4. Select **Delete endpoint**.

Troubleshoot 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.

Determine if 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.



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.

Check if 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.

Resolve 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 "InvalidAccessKeyId."

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.

Related information

- Administer StorageGRID > Troubleshoot platform services
- Create platform services endpoint

- Test connection for platform services endpoint
- Edit platform services endpoint

Configure CloudMirror replication

To enable CloudMirror replication for a bucket, you create and apply a valid bucket replication configuration XML.

Before you begin

- Platform services were enabled for your tenant account by a StorageGRID administrator.
- You have already created a bucket to act as the replication source.
- The endpoint that you intend to use as a destination for CloudMirror replication already exists, and you have its URN.
- You belong to a user group that has the Manage all buckets or Root access permission. 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.

For general information about bucket replication and how to configure it, see Amazon Simple Storage Service (S3) documentation: Replicating objects. For information about how StorageGRID implements GetBucketReplication, DeleteBucketReplication, and PutBucketReplication, see the Operations on buckets.

> CloudMirror replication has important similarities and differences with the cross-grid replication feature. To learn more, see Compare cross-grid replication and CloudMirror replication.

Note the following requirements and characteristics when configuring CloudMirror replication:

- When you create and apply a valid bucket replication configuration XML, it 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.
- 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 aren't replicated. 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 don't 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. Select **View buckets** from the dashboard, or 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** checkbox.
6. Paste the replication configuration XML into the text box, and select **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 replication is configured correctly:
   a. Add an object to the source bucket that meets the requirements for replication as specified in the replication configuration.

      In the example shown earlier, objects that match the prefix "2020" are replicated.

   b. Confirm that the object has been replicated to the destination bucket.

For small objects, replication happens quickly.

Related information

Create platform services endpoint

Configure event notifications

You enable notifications for a bucket by creating notification configuration XML and using the Tenant Manager to apply the XML to a bucket.

Before you begin

- Platform services were enabled for your tenant account by a StorageGRID administrator.
- You 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 already exists, and you have its URN.
- You belong to a user group that has the Manage all buckets or Root access permission. These permissions override the permission settings in group or bucket policies when configuring the bucket using the Tenant Manager.

About this task

You 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 Kafka or Amazon SNS topic specified as the URN of an endpoint.

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

Note the following requirements and characteristics when configuring event notifications for a bucket:

- When you create and apply valid notification configuration XML, it must use the URN of an event notifications endpoint for each destination.
- 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 won't be included in the notification messages.
- 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 Amazon SNS or Kafka topic used as the destination endpoint.
- 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** checkbox.
6. Paste the notification configuration XML into the text box, and select **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:
   a. 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 Amazon SNS or Kafka topic.

      For example, if your destination topic is hosted on the Amazon 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":"11111111111111111"
         },
         "requestParameters":{
            "sourceIPAddress":"193.51.100.20"
         },
         "responseElements":{
            "x-amz-request-id":"122047343"
         },
         "s3":{
            "s3SchemaVersion":"1.0",
            "configurationId":"Image-created",
            "bucket":{
               "name":"test1",
               "ownerIdentity":{
                  "principalId":"11111111111111111"
               },
               "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

Understand notifications for buckets

Use S3 REST API

Create platform services endpoint

Configure the search integration service

You enable search integration for a bucket by creating search integration XML and using the Tenant Manager to apply the XML to the bucket.

Before you begin

- Platform services were enabled for your tenant account by a StorageGRID administrator.
- You 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 already exists, and you have its URN.
- You belong to a user group that has the Manage all buckets or Root access permission. 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 aren't automatically sent for existing objects. Update these existing objects to ensure that their metadata is added to the destination search index.

Steps

1. Enable search integration for a bucket:
   - Use a text editor to create the metadata notification XML required to enable search integration.
   - When configuring the XML, use the URN of a search integration endpoint as the destination.

     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 aren't valid, and are rejected when they're 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.

     As needed, refer to the examples for the metadata configuration XML.

     ```
     <MetadataNotificationConfiguration>
      <Rule>
         <Status>Enabled</Status>
         <Prefix></Prefix>
         <Destination>
            <Urn>/Urn>
            </Destination>
      </Rule>
     </MetadataNotificationConfiguration>
     ```

     Elements in the metadata notification configuration XML:

Name	Description	Required
MetadataNotificationConfiguration	Container tag for rules used to specify the objects and destination for metadata notifications. Contains one or more Rule elements.	Yes
Rule	Container tag for a rule that identifies the objects whose metadata should be added to a specified index. Rules with overlapping prefixes are rejected. Included in the MetadataNotificationConfiguration element.	Yes
ID	Unique identifier for the rule. Included in the Rule element.	No
Status	Status can be 'Enabled' or 'Disabled'. No action is taken for rules that are disabled. Included in the Rule element.	Yes
Prefix	Objects that match the prefix are affected by the rule, and their metadata is sent to the specified destination. To match all objects, specify an empty prefix. Included in the Rule element.	Yes
Destination	Container tag for the destination of a rule. Included in the Rule element.	Yes

Name	Description	Required
Urn	URN of the destination where object metadata is sent. Must be the URN of a StorageGRID endpoint with the following properties: • `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`. Endpoints are configured using the Tenant Manager or Tenant Management API. They take the following form: • `arn:aws:es:region:account-ID:domain/mydomain/myindex/mytype` • `urn:mysite:es:::mydomain/myindex/mytype` The endpoint must be configured before the configuration XML is submitted, or configuration will fail with a 404 error. URN is included in the Destination element.	Yes

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** checkbox.
6. Paste the metadata notification configuration into the text box, and select **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:
   a. 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.

   b. 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 clear the **Enable search integration** checkbox.

- If you are using the S3 API directly, use a DELETE Bucket metadata notification request. See the instructions for implementing S3 client applications.

Example: 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>
```

Example: 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-1:3333333: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>
```

Metadata notification format

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"
  }
}
```

Fields included in the JSON document

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

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
```

How to view results in Elasticsearch

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. Enable the dynamic field mappings on the index before you configure the search integration service. After a document is indexed, you can't edit the document's field types in the index.

Use S3 REST API

S3 REST API supported versions and updates

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-premise object storage that uses the StorageGRID system. Minimal changes to a client application's current use of S3 REST API calls are required.

Supported versions

StorageGRID supports the following specific versions of S3 and HTTP.

Item	Version
S3 API specification	Amazon Web Services (AWS) Documentation: Amazon Simple Storage Service API Reference
HTTP	1.1 For more information about HTTP, see HTTP/1.1 (RFCs 7230-35). IETF RFC 2616: Hypertext Transfer Protocol (HTTP/1.1) **Note**: StorageGRID does not support HTTP/1.1 pipelining.

Updates to S3 REST API support

Release	Comments
11.9	• Added support for pre-calculated SHA-256 checksum values for the following requests and supported headers. You can use this feature to verify the integrity of uploaded objects: ◦ CompleteMultipartUpload: `x-amz-checksum-sha256` ◦ CreateMultipartUpload: `x-amz-checksum-algorithm` ◦ GetObject: `x-amz-checksum-mode` ◦ HeadObject: `x-amz-checksum-mode` ◦ ListParts ◦ PutObject: `x-amz-checksum-sha256` ◦ UploadPart: `x-amz-checksum-sha256` • Added the ability for the grid administrator to control tenant-level retention and Compliance settings. These settings affect S3 Object Lock settings. ◦ Bucket default retention mode and object retention mode: Governance or Compliance, if allowed by the grid administrator. ◦ Bucket default retention period and object Retain Until Date: Must be less than or equal to what is allowed by the maximum retention period set by grid administrator. • Improved support for `aws-chunked` content encoding and streaming `x-amz-content-sha256` values. Limitations: ◦ If present, `chunk-signature` is optional and not validated ◦ If present, `x-amz-trailer` content is ignored
11.8	Updated the names of S3 operations to match the names used in the Amazon Web Services (AWS) Documentation: Amazon Simple Storage Service API Reference.
11.7	• Added Quick reference: Supported S3 API requests. • Added support for using GOVERNANCE mode with S3 Object Lock. • Added support for the StorageGRID-specific `x-ntap-sg-cgr-replication-status` response header for GET Object and HEAD Object requests. This header provides an object's replication status for cross-grid replication. • SelectObjectContent requests now support Parquet objects.

Release	Comments
11.6	• Added support for using the `partNumber` request parameter in GET Object and HEAD Object requests. • Added support for a default retention mode and a default retention period at the bucket level for S3 Object Lock. • Added support for the `s3:object-lock-remaining-retention-days` policy condition key to set the range of allowable retention periods for your objects. • Changed the maximum *recommended* size for a single PUT Object operation to 5 GiB (5,368,709,120 bytes). If you have objects that are larger than 5 GiB, use multipart upload instead.
11.5	• 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 `Content-MD5` request header is now correctly supported.
11.4	• Added support for DELETE Bucket tagging, GET Bucket tagging, and PUT Bucket tagging. Cost allocation tags aren't 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 `s3:ObjectRestore:Post` 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
11.3	• 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 `x-amz-expiration` response header. • Updated PUT Object, PUT Object - Copy, and Multipart Upload to describe the impact of ILM rules that use synchronous placement at ingest. • TLS 1.1 ciphers are no longer supported.
11.2	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. **Note:** Uses of ARN/URN in other configuration JSON/XML, including those used in custom StorageGRID features, have not changed.
11.1	Added support for cross-origin resource sharing (CORS), HTTP for S3 client connections to grid nodes, and compliance settings on buckets.

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

Quick reference: Supported S3 API requests

This page summarizes how StorageGRID supports Amazon Simple Storage Service (S3) APIs.

This page includes only the S3 operations that are supported by StorageGRID.

To see the AWS documentation for each operation, select the link in the heading.

Common URI query parameters and request headers

Unless noted, the following common URI query parameters are supported:

- `versionId` (as required for object operations)

Unless noted, the following common request headers are supported:

- `Authorization`
- `Connection`
- `Content-Length`
- `Content-MD5`
- `Content-Type`
- `Date`
- `Expect`
- `Host`

- `x-amz-date`

Related information

- S3 REST API implementation details
- Amazon Simple Storage Service API Reference: Common Request Headers

AbortMultipartUpload

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus this additional URI query parameter:

- `uploadId`

Request body

None

StorageGRID documentation

Operations for multipart uploads

CompleteMultipartUpload

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus this additional URI query parameter:

- `uploadId`
- `x-amz-checksum-sha256`

Request body XML tags

StorageGRID supports these request body XML tags:

- `ChecksumSHA256`
- `CompleteMultipartUpload`
- `ETag`
- `Part`
- `PartNumber`

StorageGRID documentation

CompleteMultipartUpload

CopyObject

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional headers:

- `x-amz-copy-source`
- `x-amz-copy-source-if-match`

- `x-amz-copy-source-if-modified-since`
- `x-amz-copy-source-if-none-match`
- `x-amz-copy-source-if-unmodified-since`
- `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-metadata-directive`
- `x-amz-object-lock-legal-hold`
- `x-amz-object-lock-mode`
- `x-amz-object-lock-retain-until-date`
- `x-amz-server-side-encryption`
- `x-amz-server-side-encryption-customer-algorithm`
- `x-amz-server-side-encryption-customer-key`
- `x-amz-server-side-encryption-customer-key-MD5`
- `x-amz-storage-class`
- `x-amz-tagging`
- `x-amz-tagging-directive`
- `x-amz-meta-<metadata-name>`

Request body

None

StorageGRID documentation

CopyObject

CreateBucket

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional headers:

- `x-amz-bucket-object-lock-enabled`

Request body

StorageGRID supports all request body parameters defined by the Amazon S3 REST API at the time of implementation.

StorageGRID documentation

Operations on buckets

CreateMultipartUpload

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional headers:

- `Cache-Control`
- `Content-Disposition`
- `Content-Encoding`
- `Content-Language`
- `Expires`
- `x-amz-checksum-algorithm`
- `x-amz-server-side-encryption`
- `x-amz-storage-class`
- `x-amz-server-side-encryption-customer-algorithm`
- `x-amz-server-side-encryption-customer-key`
- `x-amz-server-side-encryption-customer-key-MD5`
- `x-amz-tagging`
- `x-amz-object-lock-mode`
- `x-amz-object-lock-retain-until-date`
- `x-amz-object-lock-legal-hold`
- `x-amz-meta-<metadata-name>`

Request body

None

StorageGRID documentation

CreateMultipartUpload

DeleteBucket

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

StorageGRID documentation

Operations on buckets

DeleteBucketCors

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

DeleteBucketEncryption

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

DeleteBucketLifecycle

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

- Operations on buckets
- Create S3 lifecycle configuration

DeleteBucketPolicy

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

DeleteBucketReplication

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

DeleteBucketTagging

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

DeleteObject

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus this additional request header:

- `x-amz-bypass-governance-retention`

Request body

None

StorageGRID documentation

Operations on objects

DeleteObjects

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus this additional request header:

- `x-amz-bypass-governance-retention`

Request body

StorageGRID supports all request body parameters defined by the Amazon S3 REST API at the time of implementation.

StorageGRID documentation

Operations on objects

DeleteObjectTagging

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on objects

GetBucketAcl

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

GetBucketCors

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

GetBucketEncryption

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

GetBucketLifecycleConfiguration

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

- Operations on buckets
- Create S3 lifecycle configuration

GetBucketLocation

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

GetBucketNotificationConfiguration

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

GetBucketPolicy

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

GetBucketReplication

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

GetBucketTagging

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

GetBucketVersioning

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

GetObject

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional URI query parameters:

- `x-amz-checksum-mode`
- `partNumber`
- `response-cache-control`
- `response-content-disposition`
- `response-content-encoding`
- `response-content-language`
- `response-content-type`
- `response-expires`

And these additional request headers:

- `Range`
- `x-amz-server-side-encryption-customer-algorithm`
- `x-amz-server-side-encryption-customer-key`
- `x-amz-server-side-encryption-customer-key-MD5`
- `If-Match`
- `If-Modified-Since`
- `If-None-Match`
- `If-Unmodified-Since`

Request body

None

StorageGRID documentation

GetObject

GetObjectAcl

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on objects

GetObjectLegalHold

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Use S3 REST API to configure S3 Object Lock

GetObjectLockConfiguration

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Use S3 REST API to configure S3 Object Lock

GetObjectRetention

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Use S3 REST API to configure S3 Object Lock

GetObjectTagging

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on objects

HeadBucket

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on buckets

HeadObject

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional headers:

- `x-amz-checksum-mode`
- `x-amz-server-side-encryption-customer-algorithm`
- `x-amz-server-side-encryption-customer-key`
- `x-amz-server-side-encryption-customer-key-MD5`
- `If-Match`
- `If-Modified-Since`
- `If-None-Match`
- `If-Unmodified-Since`
- `Range`

Request body

None

StorageGRID documentation

HeadObject

ListBuckets

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

None

StorageGRID documentation

Operations on the service > ListBuckets

ListMultipartUploads

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional parameters:

- `encoding-type`
- `key-marker`
- `max-uploads`
- `prefix`
- `upload-id-marker`

Request body

None

StorageGRID documentation

ListMultipartUploads

ListObjects

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional parameters:

- `delimiter`
- `encoding-type`
- `marker`
- `max-keys`
- `prefix`

Request body

None

StorageGRID documentation

Operations on buckets

ListObjectsV2

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional parameters:

- `continuation-token`
- `delimiter`
- `encoding-type`
- `fetch-owner`
- `max-keys`
- `prefix`

- `start-after`

Request body

None

StorageGRID documentation

Operations on buckets

ListObjectVersions

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional parameters:

- `delimiter`
- `encoding-type`
- `key-marker`
- `max-keys`
- `prefix`
- `version-id-marker`

Request body

None

StorageGRID documentation

Operations on buckets

ListParts

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional parameters:

- `max-parts`
- `part-number-marker`
- `uploadId`

Request body

None

StorageGRID documentation

ListMultipartUploads

PutBucketCors

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

StorageGRID supports all request body parameters defined by the Amazon S3 REST API at the time of

implementation.

StorageGRID documentation

Operations on buckets

PutBucketEncryption

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body XML tags

StorageGRID supports these request body XML tags:

- `ApplyServerSideEncryptionByDefault`
- `Rule`
- `ServerSideEncryptionConfiguration`
- `SSEAlgorithm`

StorageGRID documentation

Operations on buckets

PutBucketLifecycleConfiguration

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body XML tags

StorageGRID supports these request body XML tags:

- `And`
- `Days`
- `Expiration`
- `ExpiredObjectDeleteMarker`
- `Filter`
- `ID`
- `Key`
- `LifecycleConfiguration`
- `NewerNoncurrentVersions`
- `NoncurrentDays`
- `NoncurrentVersionExpiration`
- `Prefix`
- `Rule`
- `Status`

- `Tag`
- `Value`

StorageGRID documentation

- Operations on buckets
- Create S3 lifecycle configuration

PutBucketNotificationConfiguration

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body XML tags

StorageGRID supports these request body XML tags:

- `Event`
- `Filter`
- `FilterRule`
- `Id`
- `Name`
- `NotificationConfiguration`
- `Prefix`
- `S3Key`
- `Suffix`
- `Topic`
- `TopicConfiguration`
- `Value`

StorageGRID documentation

Operations on buckets

PutBucketPolicy

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

For details about the supported JSON body fields, see Use bucket and group access policies.

PutBucketReplication

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body XML tags

- `Bucket`
- `Destination`
- `Prefix`
- `ReplicationConfiguration`
- `Rule`
- `Status`
- `StorageClass`

StorageGRID documentation

Operations on buckets

PutBucketTagging

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

StorageGRID supports all request body parameters defined by the Amazon S3 REST API at the time of implementation.

StorageGRID documentation

Operations on buckets

PutBucketVersioning

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body parameters

StorageGRID supports these request body parameters:

- `VersioningConfiguration`
- `Status`

StorageGRID documentation

Operations on buckets

PutObject

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional headers:

- `Cache-Control`
- `Content-Disposition`
- `Content-Encoding`

- `Content-Language`
- `Expires`
- `x-amz-checksum-sha256`
- `x-amz-server-side-encryption`
- `x-amz-storage-class`
- `x-amz-server-side-encryption-customer-algorithm`
- `x-amz-server-side-encryption-customer-key`
- `x-amz-server-side-encryption-customer-key-MD5`
- `x-amz-tagging`
- `x-amz-object-lock-mode`
- `x-amz-object-lock-retain-until-date`
- `x-amz-object-lock-legal-hold`
- `x-amz-meta-<metadata-name>`

Request body

- Binary data of the object

StorageGRID documentation

PutObject

PutObjectLegalHold

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

StorageGRID supports all request body parameters defined by the Amazon S3 REST API at the time of implementation.

StorageGRID documentation

Use S3 REST API to configure S3 Object Lock

PutObjectLockConfiguration

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

StorageGRID supports all request body parameters defined by the Amazon S3 REST API at the time of implementation.

StorageGRID documentation

Use S3 REST API to configure S3 Object Lock

PutObjectRetention

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus this additional header:

- `x-amz-bypass-governance-retention`

Request body

StorageGRID supports all request body parameters defined by the Amazon S3 REST API at the time of implementation.

StorageGRID documentation

Use S3 REST API to configure S3 Object Lock

PutObjectTagging

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

StorageGRID supports all request body parameters defined by the Amazon S3 REST API at the time of implementation.

StorageGRID documentation

Operations on objects

RestoreObject

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

For details about the supported body fields, see RestoreObject.

SelectObjectContent

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request.

Request body

For details about the supported body fields, see the following:

- Use S3 Select
- SelectObjectContent

UploadPart

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional URI query parameters:

- `partNumber`
- `uploadId`

And these additional request headers:

- `x-amz-checksum-sha256`
- `x-amz-server-side-encryption-customer-algorithm`
- `x-amz-server-side-encryption-customer-key`
- `x-amz-server-side-encryption-customer-key-MD5`

Request body

- Binary data of the part

StorageGRID documentation

UploadPart

UploadPartCopy

URI query parameters and request headers

StorageGRID supports all common parameters and headers for this request, plus these additional URI query parameters:

- `partNumber`
- `uploadId`

And these additional request headers:

- `x-amz-copy-source`
- `x-amz-copy-source-if-match`
- `x-amz-copy-source-if-modified-since`
- `x-amz-copy-source-if-none-match`
- `x-amz-copy-source-if-unmodified-since`
- `x-amz-copy-source-range`
- `x-amz-server-side-encryption-customer-algorithm`
- `x-amz-server-side-encryption-customer-key`
- `x-amz-server-side-encryption-customer-key-MD5`
- `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`

Request body

None

StorageGRID documentation

UploadPartCopy

Test 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.

Before you begin

- You have downloaded and installed the AWS CLI from aws.amazon.com/cli.
- Optionally, you have created a load balancer endpoint. Otherwise, you know the IP address of the Storage Node you want to connect to and the port number to use. See IP addresses and ports for client connections.
- You have created an S3 tenant account.
- You have signed in to the tenant and created an access key.

For details on these steps, see Configure client connections.

Steps

1. Configure the AWS CLI settings to use the account you created in the StorageGRID system:
   a. Enter configuration mode: `aws configure`
   b. Enter the access key ID for the account you created.
   c. Enter the 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.

   This example assumes you configured a load balancer endpoint to use IP address 10.96.101.17 and port 10443.

   ```
   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 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 values

Consistency provides a balance between the availability of the objects and the consistency of those objects across different Storage Nodes and sites. You can change the consistency 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, you can:

- Specify a consistency for each bucket.
- Specify a consistency for each API operation.
- Change the default grid-wide consistency by performing one of the following tasks:
  - In the Grid Manager, go to **CONFIGURATION** > **System** > **Storage settings** > **Default consistency**.
  - Use the `grid-config` endpoint of the Grid Management private API.

A change to the grid-wide consistency applies only to buckets created after the setting was changed. To determine the details of a change, see the audit log located at `/var/local/log` (search for **consistencyLevel**).

Consistency values

The consistency 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 for a bucket or an API operation to one of the following values:

- **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**: (Default) Provides read-after-write consistency for new objects and eventual consistency for object updates. Offers high availability and data protection guarantees. Recommended for most cases.
- **Available**: Provides eventual consistency for both new objects and object updates. For S3 buckets, use only as required (for example, for a bucket that contains log values that are rarely read, or for HEAD or GET operations on keys that don’t exist). Not supported for S3 FabricPool buckets.

Use "Read-after-new-write" and "Available" consistency

When a HEAD or GET operation uses the "Read-after-new-write" consistency, 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 value until it reaches a consistency equivalent to the behavior for strong-global.

If a HEAD or GET operation uses the "Read-after-new-write" consistency but the object does not exist, the object lookup will always reach a consistency equivalent to the behavior for strong-global. Because this consistency requires multiple copies of the object metadata to be available at each site, you can receive a high number of 500 Internal Server errors if two or more Storage Nodes at the same site are unavailable.

Unless you require consistency guarantees similar to Amazon S3, you can prevent these errors for HEAD and GET operations by setting the consistency to "Available." When a HEAD or GET operation uses the "Available" consistency, StorageGRID provides eventual consistency only. It does not retry a failed operation at increasing consistency, so it does not require that multiple copies of the object metadata be available.

Specify consistency for API operation

To set the consistency for an individual API operation,the consistency values must be supported for the operation, and you must specify the consistency in the request header. This example sets the consistency to "Strong-site" for a GetObject operation.

```
GET /bucket/object HTTP/1.1
Date: date
Authorization: authorization name
Host: host
Consistency-Control: strong-site
```

> You must use the same consistency for both the PutObject and GetObject operations.

Specify consistency for bucket

To set the consistency for bucket, you can use the StorageGRID PUT Bucket consistency request. Or you can change a bucket's consistency from the Tenant Manager.

When setting the consistency for a bucket, be aware of the following:

- Setting the consistency for a bucket determines which consistency 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 for an individual API operation overrides the consistency for the bucket.
- In general, buckets should use the default consistency, "Read-after-new-write." If requests aren't working correctly, change the application client behavior if possible. Or, configure the client to specify the consistency for each API request. Set the consistency at the bucket level only as a last resort.

How consistency and ILM rules interact to affect data protection

Both your choice of consistency and your ILM rule affect how objects are protected. These settings can interact.

For example, the consistency 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 and ingest behavior can provide better initial data protection and more predictable system responses.

The following ingest options are available for ILM rules:

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.

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.

Example of how the consistency and ILM rule can interact

Suppose you have a two-site grid with the following ILM rule and the following consistency:

- **ILM rule**: Create two object copies, one at the local site and one at a remote site. Use Strict ingest behavior.
- **consistency**: 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, 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 can't be retrieved.

The inter-relationship between consistency and ILM rules can be complex. Contact NetApp if you need assistance.

Object versioning

You can set the versioning state of a bucket if you want to retain multiple versions of each object. Enabling versioning for a bucket can help protect 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 10,000 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. When versioning is enabled for a bucket, each object added to the 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 (select **Yes** for the question, "Apply this rule to older object versions only?" in Step 1 of the Create an ILM rule wizard). 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.

Related information

- How S3 versioned objects are deleted
- ILM rules and policies for S3 versioned objects (Example 4).

Use S3 REST API to configure 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. You can specify default retention for each bucket or retention settings for each object version.

How to enable 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.

S3 Object Lock is a permanent setting that can only be enabled when you create a bucket. You can't add or disable S3 Object Lock after a bucket is created.

To enable S3 Object Lock for a bucket, use either of these methods:

- Create the bucket using the Tenant Manager. See Create S3 bucket.
- Create the bucket using a CreateBucket request with the `x-amz-bucket-object-lock-enabled` request header. See Operations on buckets.

S3 Object Lock requires bucket versioning, which is enabled automatically when the bucket is created. You can't suspend versioning for the bucket. See Object versioning.

Default retention settings for a bucket

When S3 Object Lock is enabled for a bucket, you can optionally enable default retention for the bucket and specify a default retention mode and default retention period.

Default retention mode

- In COMPLIANCE mode:
  - The object can't be deleted until its retain-until-date is reached.
  - The object's retain-until-date can be increased, but it can't be decreased.
  - The object's retain-until-date can't be removed until that date is reached.
- In GOVERNANCE mode:
  - Users with the `s3:BypassGovernanceRetention` permission can use the `x-amz-bypass-governance-retention: true` request header to bypass retention settings.
  - These users can delete an object version before its retain-until-date is reached.
  - These users can increase, decrease, or remove an object's retain-until-date.

Default retention period

Each bucket can have a default retention period specified in years or days.

How to set default retention for a bucket

To set the default retention for a bucket, use either of these methods:

- Manage bucket settings from the Tenant Manager. See Create an S3 bucket and Update S3 Object Lock default retention.
- Issue a PutObjectLockConfiguration request for the bucket to specify the default mode and default number of days or years.

PutObjectLockConfiguration

The PutObjectLockConfiguration request allows you to set and modify the default retention mode and default retention period for a bucket that has S3 Object Lock enabled. You can also remove previously configured default retention settings.

When new object versions are ingested to the bucket, the default retention mode is applied if `x-amz-object-lock-mode` and `x-amz-object-lock-retain-until-date` aren't specified. The default retention period is used to calculate the retain-until-date if `x-amz-object-lock-retain-until-date` is not specified.

If the default retention period is modified after ingest of an object version, the retain-until-date of the object version remains the same and is not recalculated using the new default retention period.

You must have the `s3:PutBucketObjectLockConfiguration` permission, or be account root, to complete this operation.

The `Content-MD5` request header must be specified in the PUT request.

Request example

This example enables S3 Object Lock for a bucket and sets the default retention mode to COMPLIANCE and the default retention period to 6 years.

```
PUT /bucket?object-lock HTTP/1.1
Accept-Encoding: identity
Content-Length: 308
Host: host
Content-MD5: request header
User-Agent: s3sign/1.0.0 requests/2.24.0 python/3.8.2
X-Amz-Date: date
X-Amz-Content-SHA256: authorization-string
Authorization: authorization-string

<ObjectLockConfiguration>
    <ObjectLockEnabled>Enabled</ObjectLockEnabled>
    <Rule>
        <DefaultRetention>
            <Mode>COMPLIANCE</Mode>
            <Years>6</Years>
        </DefaultRetention>
    </Rule>
</ObjectLockConfiguration>
```

How to determine the default retention for a bucket

To determine if S3 Object Lock is enabled for a bucket and to see the default retention mode and retention period, use either of these methods:

- View the bucket in the Tenant Manager. See View S3 buckets.
- Issue a GetObjectLockConfiguration request.

GetObjectLockConfiguration

The GetObjectLockConfiguration request allows you to determine if S3 Object Lock is enabled for a bucket and, if it is enabled, see if there is a default retention mode and retention period configured for the bucket.

When new object versions are ingested to the bucket, the default retention mode is applied if `x-amz-object-lock-mode` is not specified. The default retention period is used to calculate the retain-until-date if `x-amz-object-lock-retain-until-date` is not specified.

You must have the `s3:GetBucketObjectLockConfiguration` permission, or be account root, to complete this operation.

Request example

```
GET /bucket?object-lock HTTP/1.1
Host: host
Accept-Encoding: identity
User-Agent: aws-cli/1.18.106 Python/3.8.2 Linux/4.4.0-18362-Microsoft
botocore/1.17.29
x-amz-date: date
x-amz-content-sha256: authorization-string
Authorization: authorization-string
```

Response example

```
HTTP/1.1 200 OK
x-amz-id-2:
iVmcB7OXXJRkRH1FiVq1151/T24gRfpwpuZrEG11Bb9ImOMAAe98oxSpXlknabA0LTvBYJpSIX
k=
x-amz-request-id: B34E94CACB2CEF6D
Date: Fri, 04 Sep 2020 22:47:09 GMT
Transfer-Encoding: chunked
Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<ObjectLockConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <ObjectLockEnabled>Enabled</ObjectLockEnabled>
    <Rule>
        <DefaultRetention>
            <Mode>COMPLIANCE</Mode>
            <Years>6</Years>
        </DefaultRetention>
    </Rule>
</ObjectLockConfiguration>
```

How to specify retention settings for an object

A bucket with S3 Object Lock enabled can contain a combination of objects with and without S3 Object Lock retention settings.

Object-level retention settings are specified using the S3 REST API. The retention settings for an object override any default retention settings for the bucket.

You can specify the following settings for each object:

- **Retention mode**: Either COMPLIANCE or GOVERNANCE.
- **Retain-until-date**: A date specifying how long the object version must be retained by StorageGRID.
  - In COMPLIANCE mode, if the retain-until-date is in the future, the object can be retrieved, but it can't be modified or deleted. The retain-until-date can be increased, but this date can't be decreased or removed.

    - In GOVERNANCE mode, users with special permission can bypass the retain-until-date setting. They can delete an object version before its retention period has elapsed. They can also increase, decrease, or even remove the retain-until-date.
- **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.

  The legal hold setting for an object is independent of the retention mode and the retain-until-date. If an object version is under a legal hold, no one can delete that version.

To specify S3 Object Lock settings when adding an object version to a bucket, issue a PutObject, CopyObject, or CreateMultipartUpload request.

You can use the following:

- `x-amz-object-lock-mode`, which can be COMPLIANCE or GOVERNANCE (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 aren't 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 PutObject request. `Content-MD5` is not required for CopyObject or CreateMultipartUpload.
- 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 PutObject 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 HeadObject 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.

You can use the `s3:object-lock-remaining-retention-days` policy condition key to limit the minimum and maximum allowable retention periods for your objects.

How to update retention settings for an object

If you need to update the legal hold or retention settings for an existing object version, you can perform the following object subresource operations:

- `PutObjectLegalHold`

  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.

- `PutObjectRetention`
  - The mode value can be COMPLIANCE or GOVERNANCE (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 aren't allowed.
  - If an object version has an existing retain-until-date, you can only increase it. The new value must be in the future.

How to use GOVERNANCE mode

Users who have the `s3:BypassGovernanceRetention` permission can bypass the active retention settings of an object that uses GOVERNANCE mode. Any DELETE or PutObjectRetention operations must include the `x-amz-bypass-governance-retention:true` request header. These users can perform these additional operations:

- Perform DeleteObject or DeleteObjects operations to delete an object version before its retention period has elapsed.

  Objects that are under a legal hold can't be deleted. Legal hold must be OFF.

- Perform PutObjectRetention operations that change an object version's mode from GOVERNANCE to COMPLIANCE before the object's retention period has elapsed.

  Changing the mode from COMPLIANCE to GOVERNANCE is never allowed.

- Perform PutObjectRetention operations to increase, decrease, or remove an object version's retention period.

Related information

- Manage objects with S3 Object Lock
- Use S3 Object Lock to retain objects
- Amazon Simple Storage Service User Guide: Locking Objects

Create 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 Amazon Simple Storage Service User Guide: Object lifecycle management. Note that StorageGRID only supports Expiration actions; it does not

support Transition actions.

What 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

Each object follows the retention settings of either an S3 bucket lifecycle or an ILM policy. When an S3 bucket lifecycle is configured, the lifecycle expiration actions override the ILM policy for objects matching the bucket lifecycle filter. Objects that do not match the bucket lifecycle filter use the retention settings of the ILM policy. If an object matches a bucket lifecycle filter and no expiration actions are explicitly specified, the retention settings of the ILM policy are not used and it is implied that object versions are retained forever. See Example priorities for S3 bucket lifecycle and ILM policy.

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.

> 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:

- DeleteBucketLifecycle
- GetBucketLifecycleConfiguration
- PutBucketLifecycleConfiguration

Create 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"
        }
    ]
}
```

Apply lifecycle configuration to bucket

After you have created the lifecycle configuration file, you apply it to a bucket by issuing a PutBucketLifecycleConfiguration request.

This request applies the lifecycle configuration in the example file to objects in a bucket named `testbucket`.

```
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 GetBucketLifecycleConfiguration 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.

Validate that bucket lifecycle expiration applies to object

You can determine if an expiration rule in the lifecycle configuration applies to a specific object when issuing a PutObject, HeadObject, or GetObject 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.

For example, this PutObject 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 HeadObject 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": {}
}
```

For versioning-enabled buckets, the `x-amz-expiration` response header applies only to current versions of objects.

Recommendations for implementing 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 don't expect the object to actually exist, you should use the "Available" consistency. For example, you should use the "Available" consistency 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 two or more Storage Nodes at the same site are unavailable or a remote site is unreachable.

You can set the "Available" consistency for each bucket using the PUT Bucket consistency request, or you can specify the consistency in the request header for an individual API operation.

Recommendations for object keys

Follow these recommendations for object key names, based on when the bucket was first created.

Buckets created in StorageGRID 11.4 or earlier

- Don't use random values as the first four characters of object keys. This is in contrast to the former AWS recommendation for key prefixes. Instead, 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, 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`

Buckets created in StorageGRID 11.4 or later

Restricting object key names to meet performance best practices is not required. In most cases, you can use random values for the first four characters of object key names.

> An exception to this is an S3 workload that continuously removes all objects after a short period of time. To minimize the performance impact for this use case, vary a leading portion of the key name every several thousand objects with something like the date. For example, suppose an S3 client typically writes 2,000 objects/second and the ILM or bucket lifecycle policy removes all objects after three days. To minimize the performance impact, you might name keys using a pattern like this: `/mybucket/mydir/yyyymmddhhmmss-random_UUID.jpg`

Recommendations for "range reads"

If the global option to compress stored objects is enabled, S3 client applications should avoid performing GetObject 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. GetObject operations that request a small range of bytes from a very large object are especially inefficient; for example, it is 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.

Support for Amazon S3 REST API

S3 REST API implementation details

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.

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 the common request headers defined by Amazon Simple Storage Service API Reference: Common Request Headers, with one exception.

Request header	Implementation
Authorization	Full support for AWS Signature Version 2 Support for AWS Signature Version 4, with the following exceptions: • When you provide the actual payload checksum value in `x-amz-content-sha256`, the value is accepted without validation, as if the value `UNSIGNED-PAYLOAD` had been provided for the header. When you provide an `x-amz-content-sha256` header value that implies `aws-chunked` streaming (for example, STREAMING-AWS4-HMAC-SHA256-PAYLOAD), the chunk signatures aren't verified against the chunk data.
x-amz-security-token	Not implemented. Returns `XNotImplemented`.

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

Authenticate 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.

Use 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.

Use 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 don't

need to know the secret access key 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
ListBuckets (previously named GET Service)	Implemented with all Amazon S3 REST API behavior. Subject to change without notice.
GET Storage Usage	The StorageGRID 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 / and a custom query parameter (`?x-ntap-sg-usage`) added.
OPTIONS /	Client applications can issue `OPTIONS /` 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.

Operations on buckets

The StorageGRID system supports a maximum of 5,000 buckets for each S3 tenant account.

Each grid can have a maximum of 100,000 buckets.

To support 5,000 buckets, each Storage Node in the grid must have a minimum of 64 GB of RAM.

Bucket name restrictions follow the AWS US Standard region restrictions, but you should further restrict them to DNS naming conventions to support S3 virtual hosted-style requests.

See the following for more information:

- Amazon Simple Storage Service User Guide: Bucket quotas, restrictions, and limitations
- Configure S3 endpoint domain names

The ListObjects (GET Bucket) and ListObjectVersions (GET Bucket object versions) operations support StorageGRID consistency values.

You can check whether updates to last access time are enabled or disabled for individual buckets. See GET Bucket last access time.

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
CreateBucket	Creates a new bucket. By creating the bucket, you become the bucket owner. • Bucket names must comply with the following 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. • By default, buckets are created in the `us-east-1` region; however, you can use the `LocationConstraint` request element in the request body to specify a different region. When using the `LocationConstraint` 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 don't know the region name you should use. **Note**: An error will occur if your CreateBucket request uses a region that has not been defined in StorageGRID. • You can include the `x-amz-bucket-object-lock-enabled` request header to create a bucket with S3 Object Lock enabled. See Use S3 REST API to configure S3 Object Lock. You must enable S3 Object Lock when you create the bucket. You can't 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.
DeleteBucket	Deletes the bucket.
DeleteBucketCors	Deletes the CORS configuration for the bucket.
DeleteBucketEncryption	Deletes the default encryption from the bucket. Existing encrypted objects remain encrypted, but any new objects added to the bucket aren't encrypted.
DeleteBucketLifecycle	Deletes the lifecycle configuration from the bucket. See Create S3 lifecycle configuration.
DeleteBucketPolicy	Deletes the policy attached to the bucket.
DeleteBucketReplication	Deletes the replication configuration attached to the bucket.

Operation	Implementation
DeleteBucketTagging	Uses the `tagging` subresource to remove all tags from a bucket. **Caution**: If a non-default ILM policy tag is set for this bucket, there will be a `NTAP-SG-ILM-BUCKET-TAG` bucket tag with a value assigned to it. Do not issue a DeleteBucketTagging request if there is a `NTAP-SG-ILM-BUCKET-TAG` bucket tag. Instead, issue a PutBucketTagging request with only the `NTAP-SG-ILM-BUCKET-TAG` tag and its assigned value to remove all other tags from the bucket. Do not modify or remove the `NTAP-SG-ILM-BUCKET-TAG` bucket tag.
GetBucketAcl	Returns a positive response and the ID, DisplayName, and Permission of the bucket owner, indicating that the owner has full access to the bucket.
GetBucketCors	Returns the `cors` configuration for the bucket.
GetBucketEncryption	Returns the default encryption configuration for the bucket.
GetBucketLifecycleConfiguration (previously named GET Bucket lifecycle)	Returns the lifecycle configuration for the bucket. See Create S3 lifecycle configuration.
GetBucketLocation	Returns the region that was set using the `LocationConstraint` element in the CreateBucket request. If the bucket's region is `us-east-1`, an empty string is returned for the region.
GetBucketNotificationConfiguration (previously named GET Bucket notification)	Returns the notification configuration attached to the bucket.
GetBucketPolicy	Returns the policy attached to the bucket.
GetBucketReplication	Returns the replication configuration attached to the bucket.
GetBucketTagging	Uses the `tagging` subresource to return all tags for a bucket. **Caution**: If a non-default ILM policy tag is set for this bucket, there will be a `NTAP-SG-ILM-BUCKET-TAG` bucket tag with a value assigned to it. Do not modify or remove this tag.

Operation	Implementation
GetBucketVersioning	This implementation uses the `versioning` subresource to return the versioning state of a bucket. • *blank*: Versioning has never been enabled (bucket is "Unversioned") • Enabled: Versioning is enabled • Suspended: Versioning was previously enabled and is suspended
GetObjectLockConfiguration	Returns the bucket default retention mode and default retention period, if configured. See Use S3 REST API to configure S3 Object Lock.
HeadBucket	Determines if a bucket exists and you have permission to access it. This operation returns: • `x-ntap-sg-bucket-id`: The UUID of the bucket in UUID format. • `x-ntap-sg-trace-id`: The unique trace ID of the associated request.
ListObjects and ListObjectsV2 (previously named GET Bucket)	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 `REDUCED_REDUNDANCY` storage class option: • `STANDARD`, which indicates the object is stored in a storage pool consisting of Storage Nodes. • `GLACIER`, which indicates that the object has been moved to the external bucket specified by the Cloud Storage Pool. If the bucket contains large numbers of deleted keys that have the same prefix, the response might include some `CommonPrefixes` that don't contain keys.
ListObjectVersions (previously named GET Bucket Object versions)	With READ access on a bucket, using this operation with the `versions` subresource lists metadata of all of the versions of objects in the bucket.
PutBucketCors	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 `images` to store graphics. By setting the CORS configuration for the `images` bucket, you can allow the images in that bucket to be displayed on the website `http://www.example.com`.

Operation	Implementation
PutBucketEncryption	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 `SSEAlgorithm` parameter to `AES256`, and don't use the `KMSMasterKeyID` parameter. Bucket default encryption configuration is ignored if the object upload request already specifies encryption (that is, if the request includes the `x-amz-server-side-encryption-*` request header).
PutBucketLifecycleConfiguration (previously named PUT Bucket lifecycle)	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: • Expiration (Days, Date, ExpiredObjectDeleteMarker) • NoncurrentVersionExpiration (NewerNoncurrentVersions, NoncurrentDays) • Filter (Prefix, Tag) • Status • ID StorageGRID does not support these actions: • AbortIncompleteMultipartUpload • Transition See Create S3 lifecycle configuration. To understand how the Expiration action in a bucket lifecycle interacts with ILM placement instructions, see How ILM operates throughout an object's life. **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.

Operation	Implementation

PutBucketNotificationConfiguration
 	
(previously named PUT Bucket notification)
	Configures notifications for the bucket using the notification configuration XML included in the request body. You should be aware of the following implementation details:
	
	• StorageGRID supports Amazon Simple Notification Service (Amazon SNS) or Kafka topics as destinations. Simple Queue Service (SQS) or Amazon Lambda endpoints aren't 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.
	
	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 `InvalidArgument`.
	
	• You can't 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 don't include some keys and use specific values for others, as shown in the following list:
	
	◦ **eventSource**
	
	`sgws:s3`
	
	◦ **awsRegion**
	
	not included
	
	◦ **x-amz-id-2**
	
	not included
	
	◦ **arn**
	
	`urn:sgws:s3:::bucket_name`

PutBucketPolicy	Sets the policy attached to the bucket. See Use bucket and group access policies.

Operation	Implementation
PutBucketReplication	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: • 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. For details, see Amazon Simple Storage Service User Guide: Replication configuration. • 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. See Configure CloudMirror replication. 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: URN`. • You don't need to specify a `Role` 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 `STANDARD` 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: ◦ If you delete the object or bucket before it has been replicated, the object/bucket is not replicated and you aren't 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
PutBucketTagging	Uses the `tagging` subresource to add or update a set of tags for a bucket. When adding bucket tags, be aware of the following limitations: • 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. **Caution**: If a non-default ILM policy tag is set for this bucket, there will be a `NTAP-SG-ILM-BUCKET-TAG` bucket tag with a value assigned to it. Make sure that the `NTAP-SG-ILM-BUCKET-TAG` bucket tag is included with the assigned value in all PutBucketTagging requests. Do not modify or remove this tag. **Note**: This operation will overwrite any current tags the bucket already has. If any existing tags are omitted from the set, those tags will be removed for the bucket.
PutBucketVersioning	Uses the `versioning` subresource to set the versioning state of an existing bucket. You can set the versioning state with one of the following values: • 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 `null`.
PutObjectLockConfiguration	Configures or removes the bucket default retention mode and default retention period. If the default retention period is modified, the retain-until-date of existing object versions remains the same and is not recalculated using the new default retention period. See Use S3 REST API to configure S3 Object Lock for detailed information.

Operations on objects

Operations on objects

This section describes how the StorageGRID system implements S3 REST API operations for objects.

The following conditions apply to all object operations:

- StorageGRID consistency values are supported by all operations on objects, with the exception of the following:
  - GetObjectAcl
  - `OPTIONS /`

    - PutObjectLegalHold
    - PutObjectRetention
    - SelectObjectContent
- 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 can't be accessed through S3.

The following table describes how StorageGRID implements S3 REST API object operations.

Operation	Implementation
DeleteObject	Multi-Factor Authentication (MFA) and the response header `x-amz-mfa` aren't supported. When processing a DeleteObject 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 can't 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. **Versioning** To remove a specific version, the requestor must be the bucket owner and use the `versionId` subresource. Using this subresource permanently deletes the version. If the `versionId` corresponds to a delete marker, the response header `x-amz-delete-marker` is returned set to `true`. • If an object is deleted without the `versionId` subresource on a bucket with versioning enabled, it results in the generation of a delete marker. The `versionId` for the delete marker is returned using the `x-amz-version-id` response header, and the `x-amz-delete-marker` response header is returned set to `true`. • If an object is deleted without the `versionId` subresource on a bucket with versioning suspended, 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 `x-amz-delete-marker` response header is returned set to `true`. **Note**: In certain cases, multiple delete markers might exist for an object. See Use S3 REST API to configure S3 Object Lock to learn how to delete object versions in GOVERNANCE mode.

Operation	Implementation
DeleteObjects (previously named DELETE Multiple Objects)	Multi-Factor Authentication (MFA) and the response header `x-amz-mfa` aren't supported. Multiple objects can be deleted in the same request message. See Use S3 REST API to configure S3 Object Lock to learn how to delete object versions in GOVERNANCE mode.
DeleteObjectTagging	Uses the `tagging` subresource to remove all tags from an object. **Versioning** If the `versionId` 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 `x-amz-delete-marker` response header set to `true`.
GetObject	GetObject
GetObjectAcl	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.
GetObjectLegalHold	Use S3 REST API to configure S3 Object Lock
GetObjectRetention	Use S3 REST API to configure S3 Object Lock
GetObjectTagging	Uses the `tagging` subresource to return all tags for an object. **Versioning** If the `versionId` 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 `x-amz-delete-marker` response header set to `true`.
HeadObject	HeadObject
RestoreObject	RestoreObject
PutObject	PutObject

Operation	Implementation
CopyObject (previously named PUT Object - Copy)	CopyObject
PutObjectLegalHold	Use S3 REST API to configure S3 Object Lock
PutObjectRetention	Use S3 REST API to configure S3 Object Lock
PutObjectTagging	Uses the `tagging` subresource to add a set of tags to an existing object. **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. **Tag updates and ingest behavior** When you use PutObjectTagging 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. This means that if the ILM rule uses the Strict option for ingest behavior, no action is taken if the required object placements can't be made (for example, because a newly required location is unavailable). The updated object retains its current placement until the required placement is possible. **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. **Versioning** If the `versionId` 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 `x-amz-delete-marker` response header set to `true`.
SelectObjectContent	SelectObjectContent

StorageGRID supports the following Amazon S3 Select clauses, data types, and operators for the SelectObjectContent command.

Any items not listed aren't supported.

For syntax, see SelectObjectContent. For more information about S3 Select, see the AWS documentation for S3 Select.

Only tenant accounts that have S3 Select enabled can issue SelectObjectContent queries. See the considerations and requirements for using S3 Select.

Clauses

- SELECT list
- FROM clause
- WHERE clause
- LIMIT clause

Data types

- bool
- integer
- string
- float
- decimal, numeric
- timestamp

Operators

Logical operators

- AND
- NOT
- OR

Comparison operators

- <
- >
- <=
- >=
- =
- =
- <>

- !=
- BETWEEN
- IN

Pattern matching operators

- LIKE
- _
- %

Unitary operators

- IS NULL
- IS NOT NULL

Math operators

- +
- -
- *
- /
- %

StorageGRID follows the Amazon S3 Select operator precedence.

Aggregate functions

- AVG()
- COUNT(*)
- MAX()
- MIN()
- SUM()

Conditional functions

- CASE
- COALESCE
- NULLIF

Conversion functions

- CAST (for supported datatype)

Date functions

- DATE_ADD
- DATE_DIFF

- EXTRACT
- TO_STRING
- TO_TIMESTAMP
- UTCNOW

String functions

- CHAR_LENGTH, CHARACTER_LENGTH
- LOWER
- SUBSTRING
- TRIM
- UPPER

Use 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.

Use 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:

- PutObject

- CopyObject
- CreateMultipartUpload

Use 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:

- GetObject
- HeadObject
- PutObject
- CopyObject
- CreateMultipartUpload
- UploadPart
- UploadPartCopy

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 cross-grid replication or CloudMirror replication is configured for the bucket, you can't ingest SSE-C objects. The ingest operation will fail.

Related information

Amazon S3 User Guide: Using server-side encryption with customer-provided keys (SSE-C)

CopyObject

You can use the S3 CopyObject request to create a copy of an object that is already stored in S3. A CopyObject operation is the same as performing GetObject followed by PutObject.

Resolve 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

The maximum *recommended* size for a single PutObject operation is 5 GiB (5,368,709,120 bytes). If you have objects that are larger than 5 GiB, use multipart upload instead.

The maximum *supported* size for a single PutObject operation is 5 TiB (5,497,558,138,880 bytes).

If you upgraded from StorageGRID 11.6 or earlier, the S3 PUT Object size too large alert will be triggered if you attempt to upload an object that exceeds 5 GiB. If you have a new installation of StorageGRID 11.7 or 11.8, the alert won't be triggered in this case. However, to align with the AWS S3 standard, future releases of StorageGRID won't support uploads of objects larger than 5 GiB.

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`

    If a request is made without these headers, the bucket default retention settings are used to calculate the object version mode and retain-until-date. See Use S3 REST API to configure 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`

    See Request headers for server-side encryption

Unsupported request headers

The following request headers aren't supported:

- `Cache-Control`
- `Content-Disposition`

- `Content-Encoding`
- `Content-Language`
- `Expires`
- `x-amz-checksum-algorithm`

  When you copy an object, if the source object has a checksum, StorageGRID doesn't copy that checksum value to the new object. This behavior applies whether or not you try to use `x-amz-checksum-algorithm` in the object request.

- `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 uses the Dual commit or Balanced ingest option.

- `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 CopyObject

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 can't use CopyObject 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 can't 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 CopyObject 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.

> 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 for 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 CopyObject request:
  - `x-amz-server-side-encryption`

> The `server-side-encryption` value of the object can't 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.

GetObject

You can use the S3 GetObject request to retrieve an object from an S3 bucket.

GetObject and multipart objects

You can use the `partNumber` request parameter to retrieve a specific part of a multipart or segmented object. The `x-amz-mp-parts-count` response element indicates how many parts the object has.

You can set `partNumber` to 1 for both segmented/multipart objects and non-segmented/non-multipart objects; however, the `x-amz-mp-parts-count` response element is only returned for segmented or multipart objects.

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 don't return the `x-amz-missing-meta` header if the key name or value includes unprintable characters.

Supported request header

The following request header is supported:

- `x-amz-checksum-mode`: Specify `ENABLED`

  The `Range` header isn't supported with `x-amz-checksum-mode` for GetObject. When you include `Range` in the request with `x-amz-checksum-mode` enabled, StorageGRID doesn't return a checksum value in the response.

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`.

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 Use server-side encryption.

Behavior of GetObject for Cloud Storage Pool objects

If an object has been stored in a Cloud Storage Pool, the behavior of a GetObject request depends on the state of the object. See HeadObject 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, GetObject requests will attempt to retrieve data from the grid, before retrieving it from the Cloud Storage Pool.

State of object	Behavior of GetObject
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 RestoreObject 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 RestoreObject 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 GetObject 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 GetObject request might return some data but stop midway through the transfer.
- A subsequent GetObject request might return `403 Forbidden`.

GetObject and cross-grid replication

If you are using grid federation and cross-grid replication is enabled for a bucket, the S3 client can verify an object's replication status by issuing a GetObject request. The response includes the StorageGRID-specific `x-ntap-sg-cgr-replication-status` response header, which will have one of the following values:

Grid	Replication status
Source	• **COMPLETED**: The replication was successful. • **PENDING**: The object hasn't been replicated yet. • **FAILURE**: The replication failed with a permanent failure. A user must resolve the error.
Destination	**REPLICA**: The object was replicated from the source grid.

StorageGRID does not support the `x-amz-replication-status` header.

HeadObject

You can use the S3 HeadObject 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 HeadObject to determine the object's transition state.

HeadObject and multipart objects

You can use the `partNumber` request parameter to retrieve metadata for a specific part of a multipart or segmented object. The `x-amz-mp-parts-count` response element indicates how many parts the object has.

You can set `partNumber` to 1 for both segmented/multipart objects and non-segmented/non-multipart objects; however, the `x-amz-mp-parts-count` response element is only returned for segmented or multipart objects.

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 don't return the `x-amz-missing-meta` header if the key name or value includes unprintable characters.

Supported request header

The following request header is supported:

- `x-amz-checksum-mode`

  The `partNumber` parameter and `Range` header aren't supported with `x-amz-checksum-mode` for HeadObject. When you include them in the request with `x-amz-checksum-mode` enabled, StorageGRID doesn't return a checksum value in the response.

Unsupported request header

The following request header isn't 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`.

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 Use server-side encryption.

HeadObject responses for Cloud Storage Pool objects

If the object is stored in a Cloud Storage Pool, 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 HeadObject
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	`200 OK`
	
	`x-amz-storage-class: GLACIER`
	
	`x-amz-restore: ongoing-request="false", expiry-date="Sat, 23 July 20 2030 00:00:00 GMT"`
	
	Until the object is transitioned to a non-retrievable state, the value for `expiry-date` is set to some distant time in the future. The exact time of transition is not controlled by the StorageGRID system.

State of object	Response to HeadObject

Object has transitioned to non-retrievable state, but at least one copy also exists on the grid	`200 OK`
	
	`x-amz-storage-class: GLACIER`
	
	`x-amz-restore: ongoing-request="false", expiry-date="Sat, 23 July 20 2030 00:00:00 GMT"`
	
	The value for `expiry-date` is set to some distant time in the future.
	
	Note: If the copy on the grid is not available (for example, a Storage Node is down), you must issue a RestoreObject request to restore the copy from the Cloud Storage Pool before you can successfully retrieve the 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 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 HeadObject 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.

HeadObject and cross-grid replication

If you are using grid federation and cross-grid replication is enabled for a bucket, the S3 client can verify an object's replication status by issuing a HeadObject request. The response includes the StorageGRID-specific `x-ntap-sg-cgr-replication-status` response header, which will have one of the following values:

Grid	Replication status
Source	• **COMPLETED**: The replication was successful. • **PENDING**: The object hasn't been replicated yet. • **FAILURE**: The replication failed with a permanent failure. A user must resolve the error.
Destination	**REPLICA**: The object was replicated from the source grid.

> StorageGRID does not support the `x-amz-replication-status` header.

PutObject

You can use the S3 PutObject request to add an object to a bucket.

Resolve 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

The maximum *recommended* size for a single PutObject operation is 5 GiB (5,368,709,120 bytes). If you have objects that are larger than 5 GiB, use multipart upload instead.

The maximum *supported* size for a single PutObject operation is 5 TiB (5,497,558,138,880 bytes).

> If you upgraded from StorageGRID 11.6 or earlier, the S3 PUT Object size too large alert will be triggered if you attempt to upload an object that exceeds 5 GiB. If you have a new installation of StorageGRID 11.7 or 11.8, the alert won't be triggered in this case. However, to align with the AWS S3 standard, future releases of StorageGRID won't support uploads of objects larger than 5 GiB.

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:

- PutObject, CopyObject, GetObject, and HeadObject 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-checksum-sha256`
- `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 can't use both a **User defined creation time** for the Reference time and the Balanced or Strict ingest option. 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`

    If a request is made without these headers, the bucket default retention settings are used to calculate the object version mode and retain-until-date. See Use S3 REST API to configure 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`

    See Request headers for server-side encryption

Unsupported request headers

The following request headers aren't supported:

- `x-amz-acl`
- `x-amz-sdk-checksum-algorithm`
- `x-amz-trailer`
- `x-amz-website-redirect-location`

The `x-amz-website-redirect-location` header 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 ingest option, 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 don't, 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 can't 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 can't 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.

> 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 policies, and does not result in data being stored at lower levels of redundancy in the StorageGRID system.

> 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`

    When the `x-amz-server-side-encryption` header isn't included in the PutObject request, the grid-wide stored object encryption setting is omitted from the PutObject response.
- **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.

> 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 for using server-side encryption.

> 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.

Signature calculations for the Authorization header

When using the `Authorization` header to authenticate requests, StorageGRID differs from AWS in the following ways:

- StorageGRID doesn't require `host` headers to be included within `CanonicalHeaders`.
- StorageGRID doesn't require `Content-Type` to be included within `CanonicalHeaders`.
- StorageGRID doesn't require `x-amz-*` headers to be included within `CanonicalHeaders`.

> As a general best practice, always include these headers within `CanonicalHeaders` to ensure they are verified; however, if you exclude these headers, StorageGRID does not return an error.

For details, refer to Signature Calculations for the Authorization Header: Transferring Payload in a Single

Chunk (AWS Signature Version 4).

Related information

- Manage objects with ILM
- Amazon Simple Storage Service API Reference: PutObject

RestoreObject

You can use the S3 RestoreObject request to restore an object that is stored in a Cloud Storage Pool.

Supported request type

StorageGRID only supports RestoreObject 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 don't specify `versionId`, the most recent version of the object is restored

Behavior of RestoreObject on Cloud Storage Pool objects

If an object has been stored in a Cloud Storage Pool, a RestoreObject request has the following behavior, based on the state of the object. See HeadObject 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 RestoreObject request. Instead, the local copy can be retrieved directly, using a GetObject request.

State of object	Behavior of RestoreObject
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 can't change its `expiry-date`.

State of object	Behavior of RestoreObject
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 `Tier` request element to determine how long the restore job will take to finish (`Expedited`, `Standard`, or `Bulk`). If you don't specify `Tier`, the `Standard` tier is used. **Important**: If an object has been transitioned to S3 Glacier Deep Archive or the Cloud Storage Pool uses Azure Blob storage, you can't restore it using the `Expedited` tier. The following error is returned `403 Forbidden`, `InvalidTier`: `Retrieval option is not supported by this storage class.`
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 `expiry-date` by reissuing the RestoreObject request with a new value for `Days`. The restoration date is updated relative to the time of the request.

SelectObjectContent

You can use the S3 SelectObjectContent request to filter the contents of an S3 object based on a simple SQL statement.

For more information see Amazon Simple Storage Service API Reference: SelectObjectContent.

Before you begin

- The tenant account has the S3 Select permission.
- You have `s3:GetObject` permission for the object you want to query.
- The object you want to query must be in one of the following formats:
  - **CSV**. Can be used as is or compressed into GZIP or BZIP2 archives.
  - **Parquet**. Additional requirements for Parquet objects:
    - S3 Select supports only columnar compression using GZIP or Snappy. S3 Select doesn't support whole-object compression for Parquet objects.
    - S3 Select doesn't support Parquet output. You must specify the output format as CSV or JSON.
    - The maximum uncompressed row group size is 512 MB.
    - You must use the data types specified in the object's schema.
    - You can't use INTERVAL, JSON, LIST, TIME, or UUID logical types.
- Your SQL expression has a maximum length of 256 KB.

- Any record in the input or results has a maximum length of 1 MiB.

CSV request syntax example

```
POST /{Key+}?select&select-type=2 HTTP/1.1
Host: Bucket.s3.abc-company.com
x-amz-expected-bucket-owner: ExpectedBucketOwner
<?xml version="1.0" encoding="UTF-8"?>
<SelectObjectContentRequest xmlns="http://s3.amazonaws.com/doc/2006-03-
01/">
   <Expression>string</Expression>
   <ExpressionType>string</ExpressionType>
   <RequestProgress>
      <Enabled>boolean</Enabled>
   </RequestProgress>
   <InputSerialization>
      <CompressionType>GZIP</CompressionType>
      <CSV>
         <AllowQuotedRecordDelimiter>boolean</AllowQuotedRecordDelimiter>
         <Comments>#</Comments>
         <FieldDelimiter>\t</FieldDelimiter>
         <FileHeaderInfo>USE</FileHeaderInfo>
         <QuoteCharacter>'</QuoteCharacter>
         <QuoteEscapeCharacter>\\</QuoteEscapeCharacter>
         <RecordDelimiter>\n</RecordDelimiter>
      </CSV>
   </InputSerialization>
   <OutputSerialization>
      <CSV>
         <FieldDelimiter>string</FieldDelimiter>
         <QuoteCharacter>string</QuoteCharacter>
         <QuoteEscapeCharacter>string</QuoteEscapeCharacter>
         <QuoteFields>string</QuoteFields>
         <RecordDelimiter>string</RecordDelimiter>
      </CSV>
   </OutputSerialization>
   <ScanRange>
      <End>long</End>
      <Start>long</Start>
   </ScanRange>
</SelectObjectContentRequest>
```

Parquet request syntax example

```
POST /{Key+}?select&select-type=2 HTTP/1.1
Host: Bucket.s3.abc-company.com
x-amz-expected-bucket-owner: ExpectedBucketOwner
<?xml version="1.0" encoding="UTF-8"?>
<SelectObjectContentRequest xmlns=http://s3.amazonaws.com/doc/2006-03-01/>
   <Expression>string</Expression>
   <ExpressionType>string</ExpressionType>
   <RequestProgress>
      <Enabled>boolean</Enabled>
   </RequestProgress>
   <InputSerialization>
      <CompressionType>GZIP</CompressionType>
      <PARQUET>
      </PARQUET>
   </InputSerialization>
   <OutputSerialization>
      <CSV>
         <FieldDelimiter>string</FieldDelimiter>
         <QuoteCharacter>string</QuoteCharacter>
         <QuoteEscapeCharacter>string</QuoteEscapeCharacter>
         <QuoteFields>string</QuoteFields>
         <RecordDelimiter>string</RecordDelimiter>
      </CSV>
   </OutputSerialization>
   <ScanRange>
      <End>long</End>
      <Start>long</Start>
   </ScanRange>
</SelectObjectContentRequest>
```

SQL query example

This query gets the state name, 2010 populations, estimated 2015 populations, and the percentage of change from US census data. Records in the file that aren't states are ignored.

```
SELECT STNAME, CENSUS2010POP, POPESTIMATE2015, CAST((POPESTIMATE2015 -
CENSUS2010POP) AS DECIMAL) / CENSUS2010POP * 100.0 FROM S3Object WHERE
NAME = STNAME
```

The first few lines of the file to be queried, `SUB-EST2020_ALL.csv`, look like this:

```
SUMLEV,STATE,COUNTY,PLACE,COUSUB,CONCIT,PRIMGEO_FLAG,FUNCSTAT,NAME,STNAME,
CENSUS2010POP,
ESTIMATESBASE2010,POPESTIMATE2010,POPESTIMATE2011,POPESTIMATE2012,POPESTIM
ATE2013,POPESTIMATE2014,
POPESTIMATE2015,POPESTIMATE2016,POPESTIMATE2017,POPESTIMATE2018,POPESTIMAT
E2019,POPESTIMATE042020,
POPESTIMATE2020
040,01,000,00000,00000,00000,0,A,Alabama,Alabama,4779736,4780118,4785514,4
799642,4816632,4831586,
4843737,4854803,4866824,4877989,4891628,4907965,4920706,4921532
162,01,000,00124,00000,00000,0,A,Abbeville
city,Alabama,2688,2705,2699,2694,2645,2629,2610,2602,
2587,2578,2565,2555,2555,2553
162,01,000,00460,00000,00000,0,A,Adamsville
city,Alabama,4522,4487,4481,4474,4453,4430,4399,4371,
4335,4304,4285,4254,4224,4211
162,01,000,00484,00000,00000,0,A,Addison
town,Alabama,758,754,751,750,745,744,742,734,734,728,
725,723,719,717
```

AWS-CLI usage example (CSV)

```
aws s3api select-object-content --endpoint-url https://10.224.7.44:10443
--no-verify-ssl  --bucket 619c0755-9e38-42e0-a614-05064f74126d --key SUB-
EST2020_ALL.csv --expression-type SQL --input-serialization '{"CSV":
{"FileHeaderInfo": "USE", "Comments": "#", "QuoteEscapeCharacter": "\"",
"RecordDelimiter": "\n", "FieldDelimiter": ",", "QuoteCharacter": "\"",
"AllowQuotedRecordDelimiter": false}, "CompressionType": "NONE"}' --output
-serialization '{"CSV": {"QuoteFields": "ASNEEDED",
"QuoteEscapeCharacter": "#", "RecordDelimiter": "\n", "FieldDelimiter":
",", "QuoteCharacter": "\""}}' --expression "SELECT STNAME, CENSUS2010POP,
POPESTIMATE2015, CAST((POPESTIMATE2015 - CENSUS2010POP) AS DECIMAL) /
CENSUS2010POP * 100.0 FROM S3Object WHERE NAME = STNAME" changes.csv
```

The first few lines of the output file, `changes.csv`, look like this:

```
Alabama,4779736,4854803,1.570526070895965802295356898372629785 4
Alaska,710231,738430,3.970398363349389142405780654463125377 5
Arizona,6392017,6832810,6.895992297892824753125656580700583243 1
Arkansas,2915918,2979732,2.188470320495981025529524492801237894 9
California,37253956,38904296,4.429972483996062055798852610444914897 1
Colorado,5029196,5454328,8.453279609703022113276157859029554624 6
```

AWS-CLI usage example (Parquet)

```
aws s3api select-object-content  -endpoint-url https://10.224.7.44:10443
--bucket 619c0755-9e38-42e0-a614-05064f74126d --key SUB-
EST2020_ALL.parquet --expression "SELECT STNAME, CENSUS2010POP,
POPESTIMATE2015, CAST((POPESTIMATE2015 - CENSUS2010POP) AS DECIMAL) /
CENSUS2010POP * 100.0 FROM S3Object WHERE NAME = STNAME" --expression-type
'SQL' --input-serialization '{"Parquet":{}}'  --output-serialization
'{"CSV": {}}' changes.csv
```

The first few lines of the output file, changes.csv, look like this:

```
Alabama,4779736,4854803,1.5705260708959658022953568983726297854
Alaska,710231,738430,3.9703983633493891424057806544631253775
Arizona,6392017,6832810,6.8959922978928247531256565807005832431
Arkansas,2915918,2979732,2.1884703204959810255295244928012378949
California,37253956,38904296,4.4299724839960620557988526104449148971
Colorado,5029196,5454328,8.4532796097030221132761578590295546246
```

Operations for multipart uploads

Operations for multipart uploads

This section describes how StorageGRID supports operations for multipart uploads.

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 ListMultipartUploads 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. Because 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 Balanced or Strict ingest option. You should be aware of how this affects object and part placement:
  - If ILM changes while an S3 multipart upload is in progress, some parts of the object might not meet current ILM requirements when the multipart upload completes. Any part that is not placed correctly is queued for ILM re-evaluation and 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 don't 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, each 1 GB part of a 10-part multipart upload is stored at DC2 at ingest. However, 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 values.
- When an object is ingested using multipart upload, the object segmentation threshold (1 GiB) is not applied.
- 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 CreateMultipartUpload request only. To use SSE-C (server-side encryption with customer-provided keys), you specify the same three encryption key request headers in the CreateMultipartUpload request and in each subsequent UploadPart request.

Operation	Implementation
AbortMultipartUpload	Implemented with all Amazon S3 REST API behavior. Subject to change without notice.
CompleteMultipartUpload	See CompleteMultipartUpload
CreateMultipartUpload (previously named Initiate Multipart Upload)	See CreateMultipartUpload
ListMultipartUploads	See ListMultipartUploads
ListParts	Implemented with all Amazon S3 REST API behavior. Subject to change without notice.
UploadPart	See UploadPart
UploadPartCopy	See UploadPartCopy

CompleteMultipartUpload

The CompleteMultipartUpload operation completes a multipart upload of an object by assembling the previously uploaded parts.

> StorageGRID supports non-consecutive values in ascending order for the `partNumber` request parameter with CompleteMultipartUpload. The parameter can start with any value.

Resolve 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.

Supported request headers

The following request headers are supported:

- `x-amz-checksum-sha256`
- `x-amz-storage-class`

  The `x-amz-storage-class` header affects how many object copies StorageGRID creates if the matching ILM rule specifies the Dual commit or Balanced ingest option.

- `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.

Unsupported request headers

The following request headers aren't supported:

- `x-amz-sdk-checksum-algorithm`
- `x-amz-trailer`

Versioning

This operation completes a multipart upload. If versioning is enabled for a bucket, the object version is created after 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.

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.

Refer to Troubleshoot platform services.

CreateMultipartUpload

The CreateMultipartUpload (previously named 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 ingest option, 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 ingest option, 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 don't, 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 can't 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, 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 can't 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.

> 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 policies, and does not result in data being stored at lower levels of redundancy in the StorageGRID system.

> 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.

Supported request headers

The following request headers are supported:

- `Content-Type`
- `x-amz-checksum-algorithm`

  Currently, only the SHA256 value for `x-amz-checksum-algorithm` is supported.

- `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`

  If a request is made without these headers, the bucket default retention settings are used to calculate the object version retain-until-date.

  Use S3 REST API to configure 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`

  Request headers for server-side encryption

  For information about how StorageGRID handles UTF-8 characters, see PutObject.

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 CreateMultipartUpload request if you want to encrypt the object with a unique key managed by StorageGRID. Don't specify this header in any of the UploadPart requests.
  - `x-amz-server-side-encryption`
- **SSE-C**: Use all three of these headers in the CreateMultipartUpload request (and in each subsequent UploadPart 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.

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 for using server-side encryption.

Unsupported request headers

The following request header isn't supported:

- `x-amz-website-redirect-location`

  The `x-amz-website-redirect-location` header returns `XNotImplemented`.

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 CompleteMultipartUpload operation is performed.

ListMultipartUploads

The ListMultipartUploads operation lists in-progress multipart uploads for a bucket.

The following request parameters are supported:

- `encoding-type`
- `key-marker`
- `max-uploads`
- `prefix`
- `upload-id-marker`
- `Host`
- `Date`
- `Authorization`

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 CompleteMultipartUpload operation is performed.

UploadPart

The UploadPart operation uploads a part in a multipart upload for an object.

Supported request headers

The following request headers are supported:

- `x-amz-checksum-sha256`
- `Content-Length`
- `Content-MD5`

Request headers for server-side encryption

If you specified SSE-C encryption for the CreateMultipartUpload request, you must also include the following request headers in each UploadPart 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 CreateMultipartUpload request.

- `x-amz-server-side-encryption-customer-key-MD5`: Specify the same MD5 digest that you provided in the CreateMultipartUpload 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 Use server-side encryption.

If you specified a SHA-256 checksum during the CreateMultipartUpload request, you must also include the following request header in each UploadPart request:

- `x-amz-checksum-sha256`: Specify the SHA-256 checksum for this part.

Unsupported request headers

The following request headers aren't supported:

- `x-amz-sdk-checksum-algorithm`
- `x-amz-trailer`

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 CompleteMultipartUpload operation is performed.

UploadPartCopy

The UploadPartCopy operation uploads a part of an object by copying data from an existing object as the data source.

The UploadPartCopy operation is implemented with all Amazon S3 REST API behavior. Subject to change without notice.

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 CreateMultipartUpload request, you must also include the following request headers in each UploadPartCopy 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 CreateMultipartUpload request.

- `x-amz-server-side-encryption-customer-key-MD5`: Specify the same MD5 digest that you provided in the CreateMultipartUpload request.

If the source object is encrypted using a customer-provided key (SSE-C), you must include the following three headers in the UploadPartCopy 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 Use 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 CompleteMultipartUpload operation is performed.

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
IncompleteBody	400 Bad Request
InternalError	500 Internal Server Error
InvalidAccessKeyId	403 Forbidden
InvalidArgument	400 Bad Request

Name	HTTP status
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
NoSuchBucket	404 Not Found
NoSuchKey	404 Not Found
NoSuchUpload	404 Not Found
NotImplemented	501 Not Implemented

Name	HTTP status
NoSuchBucketPolicy	404 Not Found
ObjectLockConfigurationNotFoundError	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
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 custom operations

StorageGRID custom operations

The StorageGRID system supports custom operations that are added on to the S3 REST API.

The following table lists the custom operations supported by StorageGRID.

Operation	Description
GET Bucket consistency	Returns the consistency being applied to a particular bucket.
PUT Bucket consistency	Sets the consistency applied to a particular bucket.
GET Bucket last access time	Returns whether last access time updates are enabled or disabled for a particular bucket.
PUT Bucket last access time	Allows you to enable or disable last access time updates for a particular bucket.
DELETE Bucket metadata notification configuration	Deletes the metadata notification configuration XML associated with a particular bucket.
GET Bucket metadata notification configuration	Returns the metadata notification configuration XML associated with a particular bucket.
PUT Bucket metadata notification configuration	Configures the metadata notification service for a bucket.
GET Storage Usage	Tells you the total amount of storage in use by an account and for each bucket associated with the account.
Deprecated: CreateBucket with compliance settings	Deprecated and not supported: You can no longer create new buckets with Compliance enabled.
Deprecated: GET Bucket compliance	Deprecated but supported: Returns the compliance settings currently in effect for an existing legacy Compliant bucket.
Deprecated: PUT Bucket compliance	Deprecated but supported: Allows you to modify the compliance settings for an existing legacy Compliant bucket.

GET Bucket consistency

The GET Bucket consistency request allows you to determine the consistency being applied to a particular bucket.

The default consistency is 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: date
Authorization: authorization string
Host: host
```

Response

In the response XML, `<Consistency>` will return one of the following values:

Consistency	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	(Default) Provides read-after-write consistency for new objects and eventual consistency for object updates. Offers high availability and data protection guarantees. Recommended for most cases.
available	Provides eventual consistency for both new objects and object updates. For S3 buckets, use only as required (for example, for a bucket that contains log values that are rarely read, or for HEAD or GET operations on keys that don’t exist). Not supported for S3 FabricPool buckets.

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 values

PUT Bucket consistency

The PUT Bucket consistency request allows you to specify the consistency to apply to operations performed on a bucket.

The default consistency is set to guarantee read-after-write for newly created objects.

Before you begin

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	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	(Default) Provides read-after-write consistency for new objects and eventual consistency for object updates. Offers high availability and data protection guarantees. Recommended for most cases.
available	Provides eventual consistency for both new objects and object updates. For S3 buckets, use only as required (for example, for a bucket that contains log values that are rarely read, or for HEAD or GET operations on keys that don't exist). Not supported for S3 FabricPool buckets.

Note: In general, you should use the "Read-after-new-write" consistency. If requests aren't working correctly, change the application client behavior if possible. Or, configure the client to specify the consistency for each API request. Set the consistency at the bucket level only as a last resort.

Request example

```
PUT /bucket?x-ntap-sg-consistency=strong-global HTTP/1.1
Date: date
Authorization: authorization string
Host: host
```

Related information

Consistency values

GET Bucket last access time

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: date
Authorization: authorization string
Host: host
```

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

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 or from the details page for a bucket in the Tenant Manager. See Enable or disable last access time updates.

If last access time updates are disabled for a bucket, the following behavior is applied to operations on the

bucket:

- GetObject, GetObjectAcl, GetObjectTagging, and HeadObject requests don't update last access time. The object is not added to queues for information lifecycle management (ILM) evaluation.
- CopyObject and PutObjectTagging 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, CopyObject requests don't 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, CopyObject requests always update last access time. The copy of the object is added to queues for ILM evaluation.
- CompleteMultipartUpload 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: date
Authorization: authorization string
Host: host
```

This example disables last access time for a bucket.

```
PUT /bucket?x-ntap-sg-lastaccesstime=disabled HTTP/1.1
Date: date
Authorization: authorization string
Host: host
```

DELETE Bucket metadata notification configuration

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: date
Authorization: authorization string
Host: host
```

GET Bucket metadata notification configuration

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: date
Authorization: authorization string
Host: host
```

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	Container tag for rules used to specify the objects and destination for metadata notifications. Contains one or more Rule elements.	Yes
Rule	Container tag for a rule that identifies the objects whose metadata should be added to a specified index. Rules with overlapping prefixes are rejected. Included in the MetadataNotificationConfiguration element.	Yes
ID	Unique identifier for the rule. Included in the Rule element.	No
Status	Status can be 'Enabled' or 'Disabled'. No action is taken for rules that are disabled. Included in the Rule element.	Yes
Prefix	Objects that match the prefix are affected by the rule, and their metadata is sent to the specified destination. To match all objects, specify an empty prefix. Included in the Rule element.	Yes
Destination	Container tag for the destination of a rule. Included in the Rule element.	Yes

Name	Description	Required
Urn	URN of the destination where object metadata is sent. Must be the URN of a StorageGRID endpoint with the following properties: • `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`. Endpoints are configured using the Tenant Manager or Tenant Management API. They take the following form: • `arn:aws:es:_region:account-ID_:domain/mydomain/myindex/mytype` • `urn:mysite:es:::mydomain/myindex/mytype` The endpoint must be configured before the configuration XML is submitted, or configuration will fail with a 404 error. Urn is included in the Destination element.	Yes

Response example

The XML included between the `<MetadataNotificationConfiguration></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

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 aren't 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	Container tag for rules used to specify the objects and destination for metadata notifications. Contains one or more Rule elements.	Yes
Rule	Container tag for a rule that identifies the objects whose metadata should be added to a specified index. Rules with overlapping prefixes are rejected. Included in the MetadataNotificationConfiguration element.	Yes
ID	Unique identifier for the rule. Included in the Rule element.	No
Status	Status can be 'Enabled' or 'Disabled'. No action is taken for rules that are disabled. Included in the Rule element.	Yes

Name	Description	Required
Prefix	Objects that match the prefix are affected by the rule, and their metadata is sent to the specified destination. To match all objects, specify an empty prefix. Included in the Rule element.	Yes
Destination	Container tag for the destination of a rule. Included in the Rule element.	Yes
Urn	URN of the destination where object metadata is sent. Must be the URN of a StorageGRID endpoint with the following properties: • `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`. Endpoints are configured using the Tenant Manager or Tenant Management API. They take the following form: • `arn:aws:es:region:account-ID:domain/mydomain/myindex/mytype` • `urn:mysite:es:::mydomain/myindex/mytype` The endpoint must be configured before the configuration XML is submitted, or configuration will fail with a 404 error. Urn is included in the Destination element.	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: date
Authorization: authorization string
Host: host

<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: date
Authorization: authorization string
Host: host

<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>
```

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`
System metadata	size	Object size (in bytes) as visible to an HTTP client

Type	Item name	Description
System metadata	md5	Object hash
User metadata	metadata *key:value*	All user metadata for the object, as key-value pairs
Tags	tags *key:value*	All object tags defined for the object, as key-value pairs

> 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 can't edit the document's field types in the index.

Related information

Use a tenant account

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 ListBuckets 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 can't 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: date
Authorization: authorization string
Host: host
```

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 aren't added to the `ObjectCount` total.

Related information

Consistency values

Deprecated bucket requests for legacy Compliance

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 in StorageGRID 11.6. 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.

- Use S3 REST API to configure S3 Object Lock
- Manage objects with ILM
- NetApp Knowledge Base: How to manage legacy Compliant buckets in StorageGRID 11.5

Deprecated compliance requests:

- 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.

- Deprecated - GET Bucket compliance

  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.

- Deprecated - PUT Bucket compliance

  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.

Deprecated: CreateBucket 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 CreateBucket 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. See the following for more details:
>
> - Use S3 REST API to configure S3 Object Lock
> - 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 CreateBucket 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.
```

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. See the following for more details:
>
> - Use S3 REST API to configure S3 Object Lock
> - 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: date
Authorization: authorization string
Host: host
```

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: date
Connection: connection
Server: StorageGRID/11.1.0
x-amz-request-id: request ID
Content-Length: length
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.

Name	Description
LegalHold	• True: This bucket is currently under a legal hold. Objects in this bucket can't 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	• 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`.

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. See the following for more details:
>
> • Use S3 REST API to configure S3 Object Lock
>
> • 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: date
Authorization: authorization name
Host: host
Content-Length: 152

<SGCompliance>
  <RetentionPeriodMinutes>1051200</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. **Important** 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 can't decrease that value; you can only increase it.
LegalHold	• True: This bucket is currently under a legal hold. Objects in this bucket can't 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	• 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 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 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 can't achieve the **Strong-global** consistency 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.

> Never force the **Strong-site** consistency 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 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, 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, 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

Bucket and group access policies

Use 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 managed using the GetBucketPolicy, PutBucketPolicy, and DeleteBucketPolicy S3 API operations or the Tenant Manager or Tenant Management API. 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.

There is no difference in priority between group and bucket policies.

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	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. 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.
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.

Element	Description
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 aren’t supported except to set anonymous access, which grants permission to everyone. For example, you set the wildcard (*) as the Principal value.

```
"Principal":"*"
```

```
"Principal":{"AWS":"*"}
```

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:s3:::mybucket",
        "arn:aws: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.

Consistency for policies

By default, any updates you make to group policies are eventually consistent. When 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 strongly 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 be available during a site outage.

In this case, you can either set the `Consistency-Control` header in the PutBucketPolicy request, or you can use the PUT Bucket consistency request. When 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 to a different value to work around a temporary situation, be sure to set the bucket-level setting back to its original value when you are done. Otherwise, all future bucket requests will use the modified setting.

Use 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 PutBucketPolicy operation must be encoded with charset=UTF-8.

Specify 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.

Specify principals in a policy

Use the Principal element to identity 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 don’t 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.

Specify 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 PutBucketReplication and DeleteBucketReplication 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
s3:CreateBucket	CreateBucket	Yes. **Note**: Use in group policy only.
s3:DeleteBucket	DeleteBucket	
s3:DeleteBucketMetadataNotification	DELETE Bucket metadata notification configuration	Yes
s3:DeleteBucketPolicy	DeleteBucketPolicy	
s3:DeleteReplicationConfiguration	DeleteBucketReplication	Yes, separate permissions for PUT and DELETE
s3:GetBucketAcl	GetBucketAcl	
s3:GetBucketCompliance	GET Bucket compliance (deprecated)	Yes
s3:GetBucketConsistency	GET Bucket consistency	Yes
s3:GetBucketCORS	GetBucketCors	
s3:GetEncryptionConfiguration	GetBucketEncryption	
s3:GetBucketLastAccessTime	GET Bucket last access time	Yes
s3:GetBucketLocation	GetBucketLocation	

Permissions	S3 REST API operations	Custom for StorageGRID
s3:GetBucketMetadataNotification	GET Bucket metadata notification configuration	Yes
s3:GetBucketNotification	GetBucketNotificationConfiguration	
s3:GetBucketObjectLockConfiguration	GetObjectLockConfiguration	
s3:GetBucketPolicy	GetBucketPolicy	
s3:GetBucketTagging	GetBucketTagging	
s3:GetBucketVersioning	GetBucketVersioning	
s3:GetLifecycleConfiguration	GetBucketLifecycleConfiguration	
s3:GetReplicationConfiguration	GetBucketReplication	
s3:ListAllMyBuckets	• ListBuckets • GET Storage Usage	Yes, for GET Storage Usage. **Note**: Use in group policy only.
s3:ListBucket	• ListObjects • HeadBucket • RestoreObject	
s3:ListBucketMultipartUploads	• ListMultipartUploads • RestoreObject	
s3:ListBucketVersions	GET Bucket versions	
s3:PutBucketCompliance	PUT Bucket compliance (deprecated)	Yes
s3:PutBucketConsistency	PUT Bucket consistency	Yes
s3:PutBucketCORS	• DeleteBucketCors† • PutBucketCors	
s3:PutEncryptionConfiguration	• DeleteBucketEncryption • PutBucketEncryption	

Permissions	S3 REST API operations	Custom for StorageGRID
s3:PutBucketLastAccessTime	PUT Bucket last access time	Yes
s3:PutBucketMetadataNotification	PUT Bucket metadata notification configuration	Yes
s3:PutBucketNotification	PutBucketNotificationConfiguration	
s3:PutBucketObjectLockConfiguration	• CreateBucket with the `x-amz-bucket-object-lock-enabled: true` request header (also requires the s3:CreateBucket permission) • PutObjectLockConfiguration	
s3:PutBucketPolicy	PutBucketPolicy	
s3:PutBucketTagging	• DeleteBucketTagging† • PutBucketTagging	
s3:PutBucketVersioning	PutBucketVersioning	
s3:PutLifecycleConfiguration	• DeleteBucketLifecycle† • PutBucketLifecycleConfiguration	
s3:PutReplicationConfiguration	PutBucketReplication	Yes, separate permissions for PUT and DELETE

Permissions that apply to objects

Permissions	S3 REST API operations	Custom for StorageGRID
s3:AbortMultipartUpload	• AbortMultipartUpload • RestoreObject	
s3:BypassGovernanceRetention	• DeleteObject • DeleteObjects • PutObjectRetention	
s3:DeleteObject	• DeleteObject • DeleteObjects • RestoreObject	

Permissions	S3 REST API operations	Custom for StorageGRID
s3:DeleteObjectTagging	DeleteObjectTagging	
s3:DeleteObjectVersionTagging	DeleteObjectTagging (a specific version of the object)	
s3:DeleteObjectVersion	DeleteObject (a specific version of the object)	
s3:GetObject	• GetObject • HeadObject • RestoreObject • SelectObjectContent	
s3:GetObjectAcl	GetObjectAcl	
s3:GetObjectLegalHold	GetObjectLegalHold	
s3:GetObjectRetention	GetObjectRetention	
s3:GetObjectTagging	GetObjectTagging	
s3:GetObjectVersionTagging	GetObjectTagging (a specific version of the object)	
s3:GetObjectVersion	GetObject (a specific version of the object)	
s3:ListMultipartUploadParts	ListParts, RestoreObject	
s3:PutObject	• PutObject • CopyObject • RestoreObject • CreateMultipartUpload • CompleteMultipartUpload • UploadPart • UploadPartCopy	
s3:PutObjectLegalHold	PutObjectLegalHold	
s3:PutObjectRetention	PutObjectRetention	
s3:PutObjectTagging	PutObjectTagging	

Permissions	S3 REST API operations	Custom for StorageGRID
s3:PutObjectVersionTagging	PutObjectTagging (a specific version of the object)	
s3:PutOverwriteObject	• PutObject • CopyObject • PutObjectTagging • DeleteObjectTagging • CompleteMultipartUpload	Yes
s3:RestoreObject	RestoreObject	

Use 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 can't 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 can't be overwritten.
    - Any ingest operations in progress are cancelled, and an error is returned.
    - If S3 versioning is enabled, the Deny setting prevents PutObjectTagging or DeleteObjectTagging 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 can't overwrite an object's data, user-defined metadata, or object tagging. In addition, if the **Prevent client modification** checkbox is selected (**CONFIGURATION** > **Security settings** > **Network and objects**), that setting overrides the setting of the PutOverwriteObject permission.

Specify 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: {
     condition_type: {
          condition_key: condition_values
```

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.
NumericEquals	Compares a key to a numeric value based on exact matching.

Condition operators	Description
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

Condition keys	Actions	Description
aws:SourceIp	IP operators	Will compare to the IP address from which the request was sent. Can be used for bucket or object operations. **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. **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 because its validity can't be ascertained.
aws:username	Resource/Identity	Will compare to the sender's username from which the request was sent. Can be used for bucket or object operations.
s3:delimiter	s3:ListBucket and s3:ListBucketVersions permissions	Will compare to the delimiter parameter specified in a ListObjects or ListObjectVersions request.

Condition keys	Actions	Description
s3:ExistingObjectTag/<tag-key>	s3:DeleteObjectTagging s3:DeleteObjectVersionTagging s3:GetObject s3:GetObjectAcl 3:GetObjectTagging s3:GetObjectVersion s3:GetObjectVersionAcl s3:GetObjectVersionTagging s3:PutObjectAcl s3:PutObjectTagging s3:PutObjectVersionAcl s3:PutObjectVersionTagging	Will require that the existing object has the specific tag key and value.
s3:max-keys	s3:ListBucket and s3:ListBucketVersions permissions	Will compare to the max-keys parameter specified in a ListObjects or ListObjectVersions request.
s3:object-lock-remaining-retention-days	s3:PutObject	Compares to the retain-until-date specified in the `x-amz-object-lock-retain-until-date` request header or computed from the bucket default retention period to make sure that these values are within the allowable range for the following requests: • PutObject • CopyObject • CreateMultipartUpload
s3:object-lock-remaining-retention-days	s3:PutObjectRetention	Compares to the retain-until-date specified in the PutObjectRetention request to ensure that it is within the allowable range.
s3:prefix	s3:ListBucket and s3:ListBucketVersions permissions	Will compare to the prefix parameter specified in a ListObjects or ListObjectVersions request.

Condition keys	Actions	Description
s3:RequestObjectTag/<tag-key>	s3:PutObject s3:PutObjectTagging s3:PutObjectVersionTagging	Will require a specific tag key and value when the object request includes tagging.

Specify 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
`${aws:SourceIp}`	Uses the SourceIp key as the provided variable.
`${aws:username}`	Uses the username key as the provided variable.
`${s3:prefix}`	Uses the service-specific prefix key as the provided variable.
`${s3:max-keys}`	Uses the service-specific max-keys key as the provided variable.
`${*}`	Special character. Uses the character as a literal * character.
`${?}`	Special character. Uses the character as a literal ? character.
`${$}`	Special character. Uses the character as a literal $ character.

Create 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
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 description	Policy type	Amazon behavior	StorageGRID behavior
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** > **Security** > **Security settings** > **Network and objects**, and select the **Prevent client modification** checkbox.
- 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 PutObject 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 don't 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

- How StorageGRID ILM rules manage objects
- Example bucket policies
- Example group policies

- Manage objects with ILM
- Use a tenant account

Example bucket policies

Use the examples in this section to build StorageGRID access policies for buckets.

Bucket policies specify the access permissions for the bucket that the policy is attached to. You configure a bucket policy by using the S3 PutBucketPolicy API through one of these tools:

- Tenant Manager.
- AWS CLI using this command (refer to Operations on buckets):

```
> aws s3api put-bucket-policy --bucket examplebucket --policy
file://policy.json
```

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 GetObject operations on all objects in the bucket. All other operations will be denied. Note that this policy might not be particularly useful because 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 GetObject 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/*"
    }
  ]
}
```

Example group policies

Use the examples in this section to build StorageGRID access policies for groups.

Group policies specify the access permissions for the group that the policy is attached to. There is no `Principal` element in the policy because it is implicit. Group policies are configured using the Tenant Manager or the API.

Example: Set group policy using Tenant Manager

When you add or edit a group in the Tenant Manager, you can select a group policy to determine which S3 access permissions the members of this group will have. See Create groups for an S3 tenant.

- **No S3 Access**: Default option. Users in this group don't 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 can't 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 can't edit this string.
- **Ransomware Mitigation**: This sample policy applies to all buckets for this tenant. Users in this group can perform common actions, but can't permanently delete objects from buckets that have object versioning enabled.

  Tenant Manager users who have the Manage all buckets permission can override this group policy. Limit the Manage all buckets permission to trusted users, and use Multi-Factor Authentication (MFA) where available.

- **Custom**: Users in the group are granted the permissions you specify in the text box.

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}/*"
    }
  ]
}
```

S3 operations tracked in the audit logs

Audit messages are generated by StorageGRID services and stored in text log files. You can review the S3-specific audit messages in the audit log to get details about bucket and object operations.

Bucket operations tracked in the audit logs

- CreateBucket
- DeleteBucket
- DeleteBucketTagging
- DeleteObjects
- GetBucketTagging
- HeadBucket
- ListObjects
- ListObjectVersions
- PUT Bucket compliance
- PutBucketTagging
- PutBucketVersioning

Object operations tracked in the audit logs

- CompleteMultipartUpload
- CopyObject
- DeleteObject
- GetObject
- HeadObject
- PutObject
- RestoreObject
- SelectObject
- UploadPart (when an ILM rule uses Balanced or Strict ingest)
- UploadPartCopy (when an ILM rule uses Balanced or Strict ingest)

Related information

- Access audit log file
- Client write audit messages
- Client read audit messages

Use Swift REST API (end of life)

Use Swift REST API

Support for the Swift API has reached end of life and will be removed in a future release.

Swift details have been removed from this version of the doc site. See StorageGRID 11.8: Use Swift REST API.