StorageGRID® Webscale 10.3

Simple Storage Service Implementation Guide

September 2016 | 215-10825_A0
doccomments@netapp.com

NetApp®

Contents

Support for the S3 REST API

The StorageGRID Webscale system supports the storage and retrieval of objects from client applications that interface with the StorageGRID Webscale system by using the Simple Storage Service (S3) Representational State Transfer Application Programming Interface (REST API).

Support for the S3 REST API enables you to connect service-oriented applications developed for S3 web services with on-premises object storage that uses the StorageGRID Webscale system. This requires minimal changes to a client application's current use of S3 REST API calls.

Changes to S3 REST API support

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

The following table lists changes to StorageGRID Webscale system support for the S3 REST API:

Date	Release	Comments
September 2014	10.0	Initial support of the S3 REST API by the StorageGRID Webscale system. The currently supported version of the *Simple Storage Service API Reference* is 2006-03-01.
April 2015	10.1	Added support for multipart upload, virtual hosted-style requests, and v4 authentication.
December 2015	10.2	Added support for group and bucket access policies, and for multipart copy (Upload Part - Copy).
June 2016	10.3	Added support for versioning.

Supported versions

StorageGRID Webscale supports the following specific versions of S3 and HTTP.

Item	Version
S3 specification	*Simple Storage Service API Reference* 2006-03-01
HTTP	1.1 For more information about HTTP, see HTTP/1.1 (RFC 2616).

Related information

IETF RFC 2616: Hypertext Transfer Protocol (HTTP/1.1)

Amazon Web Services (AWS) Documentation: Amazon Simple Storage Service API Reference

How client applications use HTTP connections

Client applications use HTTP connections to access and communicate with the StorageGRID Webscale system.

Client applications connect directly to an API Gateway Node or Storage Node to store and retrieve objects. To load balance ingests across Storage Nodes, you can connect to an API Gateway Node, which handles the load balancing for you. Otherwise, you can connect directly to a Storage Node.

Note: IPv6 is only supported for client application connections through the API Gateway Node.

Client applications can issue “OPTIONS /” HTTPS requests to the S3 port on a Storage Node, without providing S3 authentication credentials, to determine whether the LDR Service is available. You can use this request for monitoring, or to allow external load balancers to identify when a Storage Node is down.

Setting up the connection to client applications involves the following tasks:

- Creating an S3 account
- Identifying IP addresses for API Gateway Nodes and Storage Nodes
- Identifying S3 port numbers for API Gateway Nodes and Storage Nodes
- Copying the system's certificate authority (CA) certificate for client applications that require server validation

Related information

StorageGRID Webscale 10.3 Administrator Guide

S3 accounts in StorageGRID Webscale

You can configure the StorageGRID Webscale system to accept connections from client applications originally developed to use S3 web services by creating an S3 account.

You can create, modify, and delete S3 accounts in the NMS Management Interface. The S3 account information is used in the authentication process. When you later configure an S3 client, you need to provide the access key ID and secret access key information generated when you create the account. For more information, see “Managing S3 tenant accounts” in the *Administrator Guide*.

Related information

StorageGRID Webscale 10.3 Administrator Guide

Specifying the root domain name

Specify the root domain name for your StorageGRID Webscale system to enable support of S3 virtual hosted-style requests.

Before you begin

After the root domain name is configured, the NMS UI generates Domain Name System (DNS) resource records for each Storage Node and API Gateway Node in your StorageGRID Webscale system. You can view or download the suggested list of DNS resource records.

- You must have signed in to the Grid Management Interface using a supported browser.

- To perform this task, you need specific access permissions. For details, see information about controlling system access with administration user accounts and groups.

About this task

You can select the DNS resource records for the endpoints you want to use to configure your DNS server.

Note: The local host names of the API Gateway Nodes and Storage Nodes are assumed to be the same as the node names in the grid specification file.

The generated DNS resource records include both common entries for S3 path-style requests and wildcard (*) entries for S3 virtual hosted-style requests. Your system administrator must add the appropriate entries for each endpoint, and for the types of requests the system will support. In most cases, StorageGRID Webscale should be configured to support both styles of requests.

Your system administrator must also configure a custom server certificate for the StorageGRID Webscale system with a wildcard Subject Alternative Name (SAN) for each endpoint, and possibly each IP address. These steps are required to validate the SSL certificate and verify the hostname when API client applications connect to the endpoint.

Steps

1. Select **Configuration** > **Domain Names**.
2. Click the **Configuration** tab.
3. Enter the **Root Domain Name** to use for your StorageGRID Webscale system and click **Apply Changes**.

   For example: `sg.company.com`

   The server-specific DNS entries for all Storage Nodes and API Gateway Nodes are auto-generated.
4. If you want to download a text file with the generated resource records in the required DNS zone file format, click **Download DNS Resource Records** and save the file.
5. If you want to view the suggested list of DNS resource records for S3 path-style requests and for S3 virtual hosted-style requests, click the **Overview** tab.

Related information

StorageGRID Webscale 10.3 Administrator Guide

Viewing API endpoint domain names and DNS Resource Records

You can view the API endpoint domain names and DNS resource record information to see the available addresses to which API client applications can connect. These are suggested values.

Before you begin

Domain names are only available to client applications if the Domain Name System (DNS) has been configured for the endpoint. You must have configured the root domain name for the StorageGRID Webscale system in order to see a list of API endpoint domain names.

Steps

1. Sign in to the Grid Management Interface using a supported browser.

2. Select **Configuration > Domain Names**.

   The fully qualified domain name and suggested DNS resource record information for each API service endpoint are displayed on the Overview page.

Identifying IP addresses for API Gateway Nodes and Storage Nodes

You need the grid node's IP address to connect API client applications to StorageGRID Webscale.

Steps

1. Sign in to the Grid Management Interface using a supported browser.
2. Select **Grid**.
3. In the **Grid Topology** tree, locate and expand the Storage Node or API Gateway Node to which you want to connect.

   The services for the selected grid node appear.
4. In the Storage Node or API Gateway Node, select **SSM > Resources**, and then scroll to the **Network Addresses** table.

   You can establish HTTPS connections from API client applications to any of the listed IP addresses.

Port numbers for API Gateway Nodes and Storage Nodes

API Gateway Nodes and Storage Nodes are only available for HTTP connections from client applications to the StorageGRID Webscale system on specific port numbers.

The following ports are used for client applications that interface to the StorageGRID Webscale system through S3:

Grid node	Port number
API Gateway Node (CLB S3 Port)	8082
Storage Node (LDR S3 Port)	18082

Testing your S3 REST API configuration

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

Before you begin

- You must have downloaded and installed the AWS CLI from *aws.amazon.com/cli*.
- You must have created an S3 Account in the StorageGRID Webscale system.

Steps

1. Configure the Amazon Web Services settings to use the account you created in the StorageGRID Webscale system:

   a. Enter configuration mode:

```
aws configure
```

b. Enter the AWS Access Key ID for the account you created.

c. Enter the AWS Secret Access key for the account you created.

d. Enter the default region to use, for example, us-east-1.

e. Enter the default output format to use, or press Enter to select JSON.

2. Create a bucket.

```
aws s3api --endpoint-url https://10.96.101.17:8082
--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://10.96.101.17:8082 --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://10.96.101.17:8082 --no-verify-ssl
list-objects --bucket testbucket
```

5. Delete the object.

```
aws s3api --endpoint-url https://10.96.101.17:8082 --no-verify-ssl
delete-object --bucket testbucket --key s3.pdf
```

6. Delete the bucket.

```
aws s3api --endpoint-url https://10.96.101.17:8082 --no-verify-ssl
delete-bucket --bucket testbucket
```

How StorageGRID Webscale implements the S3 REST API

A client application can use S3 REST API calls to connect to Storage Nodes and API Gateway Nodes, to create buckets, and to store and retrieve objects.

To manage these objects, the StorageGRID Webscale system uses information lifecycle management (ILM) rules.

For information about ILM rules, see the *Administrator Guide*.

Consistency guarantees and controls

StorageGRID Webscale 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 remain eventually consistent. Overwrites generally take seconds or minutes to propagate, but can take up to 15 days.

StorageGRID Webscale now also allows the user to control consistency on a per bucket and/or API request. This allows users to trade off consistency and availability as required by their application. By default, reads of non-existent objects now require certain Storage Nodes to be available. When one or more Storage Nodes are unavailable, reading some non-existent objects might fail with an HTTP 500 error. Reading with “weak” consistency will restore the previous behavior that provides availability over read-after-write consistency.

For information on configuring consistency controls for buckets, see *GET Bucket consistency request* on page 28 and *PUT Bucket consistency request* on page 29.

To set the consistency control for an individual API operation, consistency controls must be supported for the operation, and you need to specify the consistency control in the request header. This example sets the consistency control to “strong-site” for a GET Object operation.

```
GET /bucket/object HTTP/1.1
Date: Sat, 29 Nov 2015 01:02:17 GMT
Authorization: AWS 9MOYPG9ACWPAJA1S72R5:jUGbYkLdBApjCWBgK4TxvOjfock=
Host: test.com
Consistency-Control: strong-site
```

You must set the `Consistency-Control` header to one of the following values:

- all: Provides the highest consistency guarantee. All nodes receive the data immediately or the request will fail.
- strong-global: Guarantees a consistent read-after-write view for all operations across all sites.
- strong-site: Guarantees a consistent read-after-write view for all operations within a site.
- default: Eventually consistent, highly available, with data protection guarantees.
- weak: Ensures the highest availability, but provides no consistency guarantees and a reduced guarantee of data protection.

Note: The StorageGRID Webscale `Consistency-Control` header needs to be matched for both the write and read operations. For example, using "weak" to write an object and then using "strong-global" to read the same object back does not provide strong consistency across all sites.

Related information

StorageGRID Webscale 10.3 Administrator Guide

How StorageGRID Webscale ILM rules manage objects

You can create information lifecycle management (ILM) rules that manage object data ingested into the StorageGRID Webscale system from S3 REST API client applications. You use these ILM rules to determine how and where object data is stored over time.

ILM settings determine the following aspects of an object:

Geography
The location of an object's data within the StorageGRID Webscale system.

Storage grade
The type of storage used to store object data (disc or archival media).

Loss protection
How copies are made: replication, erasure coding, or both.

Retention
The changes over time to how an object's data is managed, where it is stored, and how it is protected from loss.

You can set up an ILM rule to filter and select objects based on the interface used to ingest the object. For objects ingested using S3, you can select the following metadata as filter criteria:

- Account ID
- Bucket name
- Key
- Last access time, which is updated by GET Object and HEAD Object operations.
- Object size
- User metadata

For details about ILM rules, see the *Administrator Guide*.

Related information

StorageGRID Webscale 10.3 Administrator Guide

Object versioning

You can use versioning to retain multiple versions of an object, which protects against accidental deletion of objects, and enables you to retrieve and restore earlier versions of an object.

The StorageGRID Webscale system implements versioning with support for most features, and with some limitations.

In StorageGRID Webscale, object versioning can be combined with Information Lifecycle Management (ILM), rather than Amazon's Object Lifecycle Management, which is not supported. You must explicitly enable versioning for each bucket to turn on this functionality for the bucket. Each object in your bucket is assigned a version ID which is generated by the StorageGRID Webscale system.

Multi-factor authentication (MFA) is not supported.

Note: Versioning can be enabled only on buckets created with release 10.3 or later.

ILM and versioning

ILM policies are applied at the version level of the object. Performing change operations on ILM policies does not impact previously ingested versions. The one exception is that performing the operation to re-evaluate content does impact previously ingested versions.

S3 REST API supported operations and limitations

The StorageGRID Webscale system implements the *Simple Storage Service API Reference* (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 Webscale system supports both virtual hosted-style requests and path-style requests. Support of virtual hosted-style requests requires the root domain name to be configured (see *Specifying the root domain name* on page 6).

Related information

Amazon Web Services (AWS) Documentation: Amazon Simple Storage Service API Reference

Error responses

The StorageGRID Webscale system supports all standard S3 REST API error responses that apply. In addition, the StorageGRID Webscale implementation adds a custom `XNotImplemented` response.

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
InvalidBucketName	400 Bad Request
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
InvalidURI	400 Bad Request
KeyTooLong	400 Bad Request
MalformedXML	400 Bad Request
MetadataTooLarge	400 Bad Request

Name	HTTP Status
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
NoSuchBucketPolicy	404 Not Found
PreconditionFailed	412 Precondition Failed
RequestTimeTooSkewed	403 Forbidden
SignatureDoesNotMatch	403 Forbidden
TooManyBuckets	400 Bad Request
UserKeyMustBeSpecified	400 Bad Request

StorageGRID Webscale specific error code

Error code	Description	HTTP status code
XNotImplemented	The request you provided implies functionality that is not implemented.	501 Not Implemented

Date handling

The StorageGRID Webscale implementation of the S3 REST API only supports valid HTTP date formats.

The StorageGRID Webscale 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 Webscale system supports common request headers, with a couple of exceptions.

The StorageGRID Webscale system supports all of the common request headers defined by the *Simple Storage Service API Reference*, with the following exceptions:

Request header	Implementation
Authorization	Full support for AWS Signature Version 2 Support for AWS Signature Version 4, with the following exceptions: • The SHA256 value is not calculated for the body of the request. The user submitted value is accepted without validation. • If the chunked upload option is being used in conjunction with gzip data compression, you must specify the chunking option first. The `Content-Encoding` parameter must be specified as follows: `Content-Encoding: aws-chunked, gzip`
x-amz-copy-source-server-side-encryption-customer-key	Not implemented.
x-amz-copy-source-server-side-encryption-customer-key-MD5	Not implemented.
x-amz-security-token	Not implemented. Returns `XNotImplemented`.
x-amz-server-side-encryption-customer-algorithm	Not implemented.

Common response headers

The StorageGRID Webscale system supports common response headers, with some exceptions.

The StorageGRID Webscale system supports all of the common response headers defined by the *Simple Storage Service API Reference*, with the following exceptions:

Response header	Implementation
x-amz-id-2	Not used
x-amz-expiration	Ignored

Authenticating requests

The StorageGRID Webscale 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 Webscale system supports two authentication methods: the HTTP **`Authorization`** header and using query parameters.

Using the HTTP Authorization header

The HTTP **`Authorization`** header is used by all S3 API operations except Anonymous requests where permitted by the bucket policy. The **`Authorization`** header contains all of the required signing information to authenticate a request.

Using query parameters

You can use query parameters to add authentication information to a URL. This is known as presigning the URL, which can be used to grant temporary access to specific resources. Users with the presigned URL do not need to know the secret access key in order to access the resource, which enables you to provide third-party restricted access to a resource.

Operations on the service

The following operations on the service are supported by the StorageGRID Webscale system:

Operation	Implementation
GET Service	Implemented with all Amazon S3 REST API behavior.
OPTIONS /	Client applications can issue requests to the S3 port on a Storage Node, without providing S3 authentication credentials, to determine whether the LDR Service is available.

Operations on buckets

The StorageGRID Webscale system supports a maximum of 100 buckets per S3 account.

Bucket name restrictions follow the AWS US Standard region restrictions, but we recommend further restricting to DNS naming conventions in order to support S3 virtual hosted-style requests.

The GET Bucket (List Objects) and GET Bucket versions operations support StorageGRID Webscale consistency controls. For information on using the `Consistency-Control` header, see *How StorageGRID Webscale implements the S3 REST API* on page 10.

The following table describes which operations on buckets are implemented, and how they are implemented, in the StorageGRID Webscale system.

Operation	Implementation
DELETE Bucket	Implemented with all Amazon S3 REST API behavior.
DELETE Bucket policy	If the necessary access credentials are provided for the account, this operation deletes the policy attached to the bucket. For information about the policy language supported by the StorageGRID Webscale system, see *Group and bucket access policies* on page 34.

Operation	Implementation
GET Bucket (List Objects)	The Storage Class for objects is always listed as STANDARD, even when the object was ingested with the REDUCED_REDUNDANCY storage class specified. When an object is ingested into StorageGRID Webscale with the REDUCED_REDUNDANCY storage class, it means that the object is ingested using a single-commit ingest operation. It does not result in the object being stored at lower levels of redundancy in the StorageGRID Webscale system. Requests that traverse a large number of keys require special handling: • The request might return a truncated response that contains fewer keys than the **`max-keys`** parameter value. For example, if many keys contain a slash ("/"), then a `GET /bucket/?delimiter='/'` request would traverse at least all of these keys and the response might not return all of the expected results to avoid timing out. • The request might return an empty response to avoid timing out, and contain a **`NextMarker`** value to retrieve the next set of values. To get the complete set of results, you need to continue making requests while updating the **`marker`** parameter, as you normally do with a truncated result. Always use **`NextMarker`** if it is present. In some cases, the StorageGRID Webscale implementation of the S3 REST API returns a **`NextMarker`**, when the Amazon S3 REST API would not, because it is a better marker than the last key returned. You can check whether updates to last access time are enabled or disabled for individual buckets. For more information, see *GET Bucket last access time request* on page 31.
GET Bucket acl	If the necessary access credentials are provided for the account, this operation returns a positive response and the ID, DisplayName, and Permission of the bucket owner, indicating that the owner has full access to the bucket.
GET Bucket location	If the necessary access credentials are provided for the account, this operation will return an empty string for the bucket's region. Buckets are not location constrained in StorageGRID Webscale.

Operation	Implementation
GET Bucket Object versions	With READ access on a bucket, this operation with the `versions` subresource lists metadata of all of the versions of objects in the bucket. Implementation notes for GET Bucket (List Objects) apply for GET Bucket Object versions with the following exceptions: • `key-marker` (instead of `marker` for GET Bucket operation) specifies the key in the bucket from which to start listing • `version-id-marker` specifies the version associated with the `key-marker` from which to start version listing `NextKeyMarker` and `NextVersionIdMarker` response headers specify the first object version not returned that satisfies the search criteria. These values could be used with a subsequent request to continue the listing.
GET Bucket policy	If the necessary access credentials are provided for the account, this operation returns the policy attached to the bucket. For information about the policy language supported by the StorageGRID Webscale system, see *Group and bucket access policies* on page 34.
GET Bucket versioning	This implementation uses the `versioning` subresource to return the versioning state of a bucket. To retrieve the versioning state of a bucket, you must be the bucket owner. The versioning state returned indicates if the bucket is "Unversioned" or if the bucket is version "Enabled" or "Suspended".
HEAD Bucket	Implemented with all Amazon S3 REST API behavior.
PUT Bucket	Location constraints are ignored. You can enable or disable updates to last access time for each individual bucket. For more information, see *PUT Bucket last access time request* on page 32.
PUT Bucket policy	If the necessary access credentials are provided for the account, this operation sets the policy attached to the bucket. For information about the policy language supported by the StorageGRID Webscale system, see *Group and bucket access policies* on page 34.
PUT Bucket versioning	This implementation uses the `versioning` subresource to set the versioning state of an existing bucket. To set the versioning state, you must be the bucket owner. 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`.

Operations on objects

Data objects ingested to the StorageGRID Webscale system through Swift or CDMI cannot be accessed through S3.

All of the operations on objects, except GET Object ACL and OPTIONS / support StorageGRID Webscale consistency controls. For information on using the `Consistency-Control` header, see *How StorageGRID Webscale implements the S3 REST API* on page 10.

The following operations on objects are supported by the StorageGRID Webscale system:

Operation	Implementation
DELETE Object	Multi-Factor Authentication (MFA) and the response header `x-amz-mfa` are not supported. **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 version enabled bucket, 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 version suspended bucket, it results in a permanent deletion of an already existing 'null' version or a 'null' delete marker, and the generation of a new 'null' delete marker. The `x-amz-delete-marker` response header is returned set to `true`.
DELETE Multiple Objects	Multi-Factor Authentication (MFA) and the response header `x-amz-mfa` are not supported. Multiple objects can be deleted in the same request message. **Note:** The DELETE Multiple Objects request is not supported on versioned buckets. Unlike the PUT object operation, the DELETE Multiple Objects operation does not support chunked transfer encoding and the content encoding gzip attributes.
GET Object	The following request headers are not supported and return `XNotImplemented`: • `x-amz-restore` • `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`.

Operation	Implementation
GET Object ACL	If the necessary access credentials are provided for the account, the operation returns a positive response and the ID, DisplayName, and Permission of the object owner, indicating that the owner has full access to the object.
HEAD Object	The following request headers are not supported and return `XNotImplemented`: • `x-amz-restore` • `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`.

Operation	Implementation
PUT Object	**Storage class options** The `x-amz-storage-class` request header is supported with the following enumerated values: • `STANDARD`: (Default) Specifies a dual-commit ingest operation. • `REDUCED_REDUNDANCY`: Specifies a single-commit ingest operation. **Note:** Specifying the `REDUCED_REDUNDANCY` value does not affect the specified information lifecycle management (ILM) policy, and does not result in data being stored at lower levels of redundancy in the StorageGRID Webscale system. The PUT Object operation supports the `Transfer-Encoding:chunked` and `Content-Encoding:gzip` attributes. **Request headers** The following request headers are supported: • `x-amz-server-side-encryption` • `x-amz-meta-` name-value pairs for user-defined metadata To record the object creation time, so that you can use the User Defined Creation Time option for the reference time in an ILM rule, you need to store the value in a user-defined header named `x-amz-meta-creation-time`. For example: `x-amz-meta-creation-time=1443399726`. This field is evaluated as seconds since Jan 1, 1970. For more information, see "Reference time" in the *Administrator Guide*. The following request headers are not supported and return `XNotImplemented`: • `Expires` • `x-amz-acl` • `x-amz-server-side-encryption-customer-algorithm` • `x-amz-server-side-encryption-customer-key` • `x-amz-server-side-encryption-customer-key-MD5` • `x-amz-website-redirect-location` **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. For more information on versioning, see the "PUT Bucket versioning" and "GET Bucket versioning" entries in *Operations on buckets* on page 16.

Operation	Implementation
PUT Object - Copy	The following request headers are supported: • `x-amz-meta-` name-value pairs for user-defined metadata To record the object creation time, so that you can use the User Defined Creation Time option for the reference time in an ILM rule, you need to store the value in a user-defined header named `x-amz-meta-creation-time`. For example: `x-amz-meta-creation-time`=1443399726. This field is evaluated as seconds since Jan 1, 1970. For more information, see "Reference time" in the *Administrator Guide*. • `x-amz-metadata-directive`: The default value is `COPY`, which enables you to update the object metadata and copy the object. You can specify `REPLACE` to overwrite an existing object and the associated metadata. • `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-server-side-encryption` • `x-amz-storage-class` The following request headers are not supported and return `XNotImplemented`: • `x-amz-server-side-encryption-customer-algorithm` • `x-amz-server-side-encryption-customer-key` • `x-amz-server-side-encryption-customer-key-MD5` • `x-amz-website-redirect-location` If the `x-amz-copy-source` object data matches the destination, the request updates object metadata only. If the `x-amz-copy-source` object data is different than the destination, the request reads and writes the object data within the StorageGRID Webscale system to create a copy. **Versioning** If the source bucket is versioned, you can use the `x-amz-copy-source` header to copy the latest version of an object. To copy a specific version of an object, you must explicitly specify the version to copy using the `versionId` subresource. If the destination bucket is versioned, the generated version is returned in the `x-amz-version-id` response header. If versioning is suspended for the target bucket, then `x-amz-version-id` returns a "null" value.

Related information

StorageGRID Webscale 10.3 Administrator Guide

Operations for multipart uploads

Note: You should not exceed 1,000 concurrent multipart uploads to a single bucket because the results of List Multipart Uploads queries for that bucket might return incomplete results.

Note: Each multipart part except the last must be between 5 MB to 5 GB. The last part can be less than 5 MB. This client must follow these limits as it is not enforced by StorageGRID Webscale.

All of the multipart upload operations support StorageGRID Webscale consistency controls. For information on using the `Consistency-Control` header, see *How StorageGRID Webscale implements the S3 REST API* on page 10.

Operation	Implementation
List Multipart Uploads	The following request headers are supported: • `encoding-type` • `max-uploads` • `key-marker` • `prefix` • `upload-id-marker` The `delimiter` request parameter is not supported.

Operation	Implementation
Initiate Multipart Upload	The `x-amz-storage-class` request header is supported with the following enumerated values: • `STANDARD` Default. Specifies a dual-commit ingest operation. • `REDUCED_REDUNDANCY` Specifies a single-commit ingest operation. **Note:** Specifying the `REDUCED_REDUNDANCY` value does not affect the specified ILM policy, and does not result in data being stored at lower levels of redundancy in the StorageGRID Webscale system. The following request headers are supported: • `Content-Type` • `x-amz-meta-` name-value pairs for user-defined metadata To record the object creation time, so that you can use the User Defined Creation Time option for the reference time in an ILM rule, you need to store the value in a user-defined header named `x-amz-meta-creation-time`. For example: `x-amz-meta-creation-time`=1443399726. This field is evaluated as seconds since Jan 1, 1970. For more information, see "Reference time" in the *Administrator Guide*. The `x-amz-server-side-encryption` header is not supported directly for Initiate Multipart Upload requests. If you require server-side encryption for a multipart upload, you need to specify the `x-amz-server-side-encryption` header for each of the upload parts, but you cannot specify it as part of the Initiate Multipart Upload header or the request fails. The following request headers are not supported and return `XNotImplemented`: • `x-amz-website-redirect-location` • `x-amz-server-side-encryption-customer`

Operation	Implementation
Upload Part	The following request headers are supported: • `Content-Length` • `x-amz-server-side-encryption` If you need to specify server-side encryption for a multipart upload, you need to specify the `x-amz-server-side-encryption` header for each of the individual upload parts. The following request headers are not supported and return `XNotImplemented`: • `x-amz-server-side-encryption-customer-algorithm` • `x-amz-server-side-encryption-customer-key` • `x-amz-server-side-encryption-customer-key-MD5`
Upload Part - Copy	Implemented with all Amazon S3 REST API behavior. This request reads and writes the object data specified in `x-amz-copy-source-range` within the StorageGRID Webscale system.

Operation	Implementation
Complete Multipart Upload	The `x-amz-storage-class` request header is supported with the following enumerated values: • `STANDARD` Default. Specifies a dual-commit ingest operation. • `REDUCED_REDUNDANCY` Specifies a single-commit ingest operation. **Note:** Specifying the `REDUCED_REDUNDANCY` value does not affect the specified ILM policy, and does not result in data being stored at lower levels of redundancy in the StorageGRID Webscale system. **Attention:** 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 following request headers are not supported and return `XNotImplemented`: • `x-amz-expiration` • `x-amz-server-side-encryption-customer-algorithm` • `x-amz-server-side-encryption-customer-key` • `x-amz-server-side-encryption-customer-key-MD5` • `x-amz-version-id` **Note:** 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.
Abort Multipart Upload	Implemented with all Amazon S3 REST API behavior.
List Parts	The following request parameters are supported: • `max-parts` • `part-number-marker`

Related information

StorageGRID Webscale 10.3 Administrator Guide

Operations tracked in the audit logs

Several bucket operations and object operations are tracked in the StorageGRID Webscale audit logs.

Bucket operations	Object operations
DELETE Bucket	Complete Multipart Upload
GET Bucket (List Objects)	DELETE Object

Bucket operations	**Object operations**
HEAD Bucket	GET Object
PUT Bucket	HEAD Object
PUT Bucket Versioning	PUT Object
	PUT Object - Copy

StorageGRID Webscale S3 REST API operations

There are operations added on to the S3 REST API that are specific to StorageGRID Webscale system.

GET Bucket consistency request

The GET Bucket consistency request allows you to determine the consistency level being applied to a particular bucket.

Request

Request HTTP Header	Description
Authorization	Specifies the AWS signature and the access key ID for the account to use for the request.
Date	The date and time of the request.
Host	The hostname to which the request is directed.

Request Example

```
GET /bucket?x-ntap-sg-consistency HTTP/1.1
Date: Sat, 29 Nov 2015 01:02:17 GMT
Authorization: AWS 9MOYPG9ACWPAJA1S72R5:jUGbYkLdBApjCWBgK4TxvOjfock=
Host: test.com
```

Response

Response HTTP Header	Description
Connection	Specifies whether the connection to the server is open or closed.
Content-Length	The length of the response body.
Content-Type	The Multipurpose Internet Mail Extensions (MIME) type of the response body.
Date	The date and time of the response.
Server	The server that created the response.
x-amz-request-id	The identifier that uniquely identifies the request. Created by the S3 API.

`<Consistency>` will return one of the following values:

- all: Provides the highest consistency guarantee. All nodes receive the data immediately or the request will fail.
- strong-global: Guarantees a consistent read-after-write view for all operations across all sites.
- strong-site: Guarantees a consistent read-after-write view for all operations within a site.
- default: Eventually consistent, highly available, with data protection guarantees.

- weak: Ensures the highest availability, but provides no consistency guarantees and a reduced guarantee of data protection.

Response Example

```
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"?>
<Consistency xmlns="http://s3.storagegrid.com/doc/2015-02-01/">default</
Consistency>
```

PUT Bucket consistency request

The PUT Bucket consistency request allows you to specify the consistency level to apply to operations performed on a bucket.

Request

Request HTTP Header	Description
Authorization	Specifies the AWS signature and the access key ID for the account to use for the request.
Date	The date and time of the request.
Host	The hostname to which the request is directed.

The `x-ntap-sg-consistency` parameter must contain one of the following values:

- all: Provides the highest consistency guarantee. All nodes receive the data immediately or the request will fail.
- strong-global: Guarantees a consistent read-after-write view for all operations across all sites.
- strong-site: Guarantees a consistent read-after-write view for all operations within a site.
- default: Eventually consistent, highly available, with data protection guarantees.

Request example

```
PUT /bucket?x-ntap-sg-consistency=strong-global HTTP/1.1
Date: Sat, 29 Nov 2015 01:02:17 GMT
Authorization: AWS 9MOYPG9ACWPAJA1S72R5:jUGbYkLdBApjCWBgK4TxvOjfock=
Host: test.com
```

GET Storage Usage request

The GET Storage Usage request tells you the total amount of storage in use by an account, and for each bucket associated with the account.

The amount of storage used by an account and its buckets can be obtained by a modified GET Service request with the `x-ntap-sg-usage` query parameter. Bucket storage usage is tracked separately from the PUT and DELETE requests processed by the system. There might be some delay

before the usage values match the expected values based on the processing of requests, particularly if the system is under heavy load.

Request

Request HTTP Header	Description
Authorization	Specifies the AWS signature and the access key ID for the account to use for the request.
Date	The date and time of the request.
Host	The hostname to which the request is directed.

Request example

```
GET /?x-ntap-sg-usage HTTP/1.1
Date: Sat, 29 Nov 2015 00:49:04 GMTT
Authorization: AWS 9MOYPG9ACWPAJA1S72R5:jUGbYkLdBApjCWBgK4TxvOjfock=
Host: test.com
```

Response

Response HTTP Header	Description
Connection	Specifies whether the connection to the server is open or closed.
Content-Length	The length of the response body.
Content-Type	The Multipurpose Internet Mail Extensions (MIME) type of the response body.
Date	The date and time of the response.
Server	The server that created the response.
x-amz-request-id	The identifier that uniquely identifies the request. Created by the S3 API.

Response example

```
HTTP/1.1 200 OK
Date: Sat, 29 Nov 2015 00:49:05 GMT
Connection: KEEP-ALIVE
Server: StorageGRID/10.2.0
x-amz-request-id: 727237123
Content-Length: 427
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<UsageResult xmlns="http://s3.storagegrid.com/doc/2015-02-01">
<CalculationTime>2014-11-19T05:30:11.000000Z</CalculationTime>
<ObjectCount>4</ObjectCount>
<DataBytes>12</DataBytes>
<Buckets>
<Bucket>
<Name>bucket1</Name>
<ObjectCount>2</ObjectCount>
<DataBytes>6</DataBytes>
</Bucket>
<Bucket>
<Name>bucket2</Name>
<ObjectCount>2</ObjectCount>
```

```
<DataBytes>6</DataBytes>
</Bucket>
</Buckets>
</UsageResult>
```

Versioning

Every object version stored will contribute to the `ObjectCount` and `DataBytes` values in the response. Delete markers are not added to the `ObjectCount` total.

GET Bucket last access time request

The GET Bucket last access time request allows you to determine if last access time updates are enabled or disabled for individual buckets.

You must have s3:GetBucketPolicy permissions to complete this operation.

Request

Request HTTP Header	Description
Authorization	Specifies the AWS signature and the access key ID for the account to use for the request.
Date	The date and time of the request.
Host	The hostname to which the request is directed.

Request Example

```
GET /bucket?x-ntap-sg-lastaccesstime HTTP/1.1
Date: Sat, 29 Nov 2015 01:02:17 GMT
Authorization: AWS 9MOYPG9ACWPAJA1S72R5:jUGbYkLdBApjCWBgK4TxvOjfock=
Host: test.com
```

Response

Response HTTP Header	Description
Connection	Specifies whether the connection to the server is open or closed.
Content-Length	The length of the response body.
Content-Type	The Multipurpose Internet Mail Extensions (MIME) type of the response body.
Date	The date and time of the response.
Server	The server that created the response.
x-amz-request-id	The identifier that uniquely identifies the request. Created by the S3 API.

Response Example

The value of `<LastAccessTime>` will either be enabled or disabled.

```
HTTP/1.1 200 OK
Date: Sat, 29 Nov 2015 01:02:18 GMT
Connection: CLOSE
```

```
Server: StorageGRID/10.3.0
x-amz-request-id: 12345
Content-Length: 127
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<LastAccessTime xmlns="http://s3.storagegrid.com/doc/2015-02-01/">enabled
</LastAccessTime>
```

PUT Bucket last access time request

The PUT Bucket last access time request allows you to enable or disable last access time updates for individual buckets. Disabling last access time improves performance, and is the default setting for all buckets created with version 10.3.0, or later.

You must have s3:PutBucketPolicy permissions for a bucket to enable or disable the last access time configuration settings for the bucket.

> **Note:** Last access time updates are enabled by default for all buckets created with versions prior to 10.3.0. You need to explicitly disable last access time for each of these buckets to match the default behavior of version 10.3.0, or later. You can only change the last access time configuration settings for a bucket using this S3 API operation.

If last access time is disabled for a bucket the following behavior is applied to operations on the bucket:

- GET Object, GET Object ACL, and HEAD Object do not update last access time, and are not added to queues for information lifecycle management (ILM) evaluation.
- PUT Object Copy that only updates metadata does not update last access time, but is added to queues for ILM evaluation.
- PUT Object Copy and Complete Multipart Upload do update last access time, and are added to queues for ILM evaluation.

Request

Request HTTP Header	Description
Authorization	Specifies the AWS signature and the access key ID for the account to use for the request.
Date	The date and time of the request.
Host	The hostname to which the request is directed.

Request examples

Enabling last access time for a bucket

```
PUT /bucket?x-ntap-sg-lastaccesstime=enabled HTTP/1.1
Date: Sat, 29 Nov 2015 01:02:17 GMT
Authorization: AWS 9MOYPG9ACWPAJA1S72R5:jUGbYkLdBApjCWBgK4TxvOjfock=
Host: test.com
```

Disabling last access time for a bucket

```
PUT /bucket?x-ntap-sg-lastaccesstime=disabled HTTP/1.1
Date: Sat, 29 Nov 2015 01:02:17 GMT
Authorization: AWS 9MOYPG9ACWPAJA1S72R5:jUGbYkLdBApjCWBgK4TxvOjfock=
Host: test.com
```

Configuring security for the REST API

You need to understand the security measures implemented for the REST API and how to secure your system.

How the StorageGRID Webscale system implements security for the REST API

The StorageGRID Webscale system employs the use of Transport Layer Security (TLS) connection security, server authentication, client authentication, and client authorization. When considering security issues, you might find it helpful to understand how the StorageGRID Webscale system implements security, authentication, and authorization for the S3 or Swift REST API.

The StorageGRID Webscale system accepts HTTPS commands submitted over a network connection that uses TLS to provide connection security, application authentication and, optionally, transport encryption. Commands that do not use TLS are rejected. If an object is encrypted when it is ingested, it stays encrypted for the lifetime of the object in the StorageGRID Webscale system.

TLS enables the exchange of certificates as entity credentials and allows a negotiation that can use hashing and encryption algorithms.

When a StorageGRID Webscale system is installed, a certificate authority (CA) certificate is generated for the system, as well as server certificates for each Storage Node. These server certificates are all signed by the system CA. You need to configure client applications to trust this CA certificate. When a client application connects to any Storage Node using TLS, the application can authenticate the Storage Node by verifying that the server certificate presented by the Storage Node is signed by the trusted system CA.

Alternatively, you can choose to supply a single, custom server certificate that should be used on all Storage Nodes rather than the generated ones. The custom server certificate must be signed by a CA selected by the administrator. The server authentication process by the client application is the same, but in this instance with a different trusted CA. For more information, see “Configuring certificates” in the *Administrator Guide*.

The following table shows how security issues are implemented for S3 and Swift API:

Security issue	Implementation for REST API
Connection security	TLS
Server authentication	X.509 server certificate signed by system CA or custom server certificate supplied by administrator
Client authentication	• S3: S3 account (access key ID and secret access key) • Swift: Swift account (credentials of user name and password)
Client authorization	• S3: Bucket ownership and all applicable access control policies • Swift: Account admin role access

Related information

StorageGRID Webscale 10.3 Administrator Guide

Group and bucket access policies

The StorageGRID Webscale system implements a subset of the S3 API policy language that you can use to control access to buckets and objects within those buckets.

Overview

StorageGRID Webscale bucket and group policies contain statements. Statements contain the following elements, which you need to define:

- Resources
  You can allow or deny permissions to buckets and objects using the uniform resource name (URN) to identify the resource.
- Principals
  You can allow 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 (*). Access to resources is granted to anonymous users through the permissions. By default, anonymous users have no access to resources.
  You only need to specify the principal in a bucket policy. For group policies, the group to which the policy is attached is the implicit principal.
- 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 also explicitly deny access to a resource, so that a group cannot access it even if a different policy grants access. Permissions have two components:
  - Action
    You need to identify operations you allow (or deny) on buckets or objects using the supported action keywords. You can use the wildcard character (*) to specify all operations, or a subset of operations (i.e. `"s3:*Object"`).
  - Effect
    You need to specify whether the specified operations are allowed or denied.

The following example policy shows a complete bucket policy that allows the admin and finance groups `s3:ListBucket` and `s3:GetObject` permissions for the `mybucket` bucket:

```
{
"Statement": [
  {
    "Effect": "Allow",
    "Principal": {
      "SGWS": [
         "urn:sgws:identity::27233906934684427525:federated-group/admin",
         "urn:sgws:identity::27233906934684427525:federated-group/
finance"]
    },
    "Action": ["s3:ListBucket","s3:GetObject"],
    "Resource": ["urn:sgws:s3:::mybucket", "urn:sgws: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.

Group policies are cached for 15 minutes, and bucket policies are cached for 8 seconds. Therefore, due to caching, changes to group and bucket policies may take up to 15 minutes to take effect across

all grid nodes. If you want bucket policy changes to take effect immediately, you can set the bucket consistency level to “all” using the PUT bucket consistency request or set the “Consistency-Control” header to “all” for the PUT bucket policy request.

Specify resources in a policy

You use the common uniform resource name (URN) format to identify any S3 resources or identity resources in the StorageGRID Webscale system:

```
urn:sgws:s3:::bucket_name
urn:sgws:s3:::bucket_name/key_name
```

```
urn:sgws:identity::27233906934684427525:root
urn:sgws:identity::27233906934684427525:user/Bob
```

- The StorageGRID Webscale REST API implementation of identity resources differs from Amazon's implementation:
  - For principals, the service component is “identity” instead of “iam”.
  - For principals, the `group-uuid` resource type is an additional StorageGRID Webscale specific resource type.
    You need to specify the resource type and the UUID, instead of using a UUID alone.
    For example:

    ```
    urn:sgws:identity::27233906934684427525:group-uuid/
    de305d54-75b4-431b-adb2-eb6b9e546013
    ```

  - For resources, the region component must be empty.
  - For resources, service component remains “s3”.
    For example:

    ```
    "Resource": "urn:sgws:s3:::mybucket/*"
    ```

- The principal value can specify a group name that does not yet exist when the bucket policy is created.
- The resource value can specify a bucket that does not yet exist when the group policy is created.
- The version value is not used; if you specify one, it is ignored and the following apply:
  - Policy variables are not supported.
  - Conditions are not supported.
- International characters, which can be specified in the object key, should be encoded using JSON UTF-8 or use JSON \u escape sequences.
  Percent-encoding, as outlined in *RFC 2141 URN Syntax*, is not supported. The request body for the PUT Bucket policy operation must be encoded as JSON UTF-8, and UTF-8 is always set as the content type.

Specify a principal in a policy

Account-based identities must be specified in one of the following formats:

- `"SGWS": "`*`account_ID`*`"`
- `"SGWS": "`*`account_URN`*`"`

You can specify an account using an ID. This example uses the ID 27233906934684427525, which includes the account root and all users in the account):

```
 "Principal": { "SGWS": "27233906934684427525" }
```

You can specify just the account root:

```
"Principal": { "SGWS": "urn:sgws:identity::27233906934684427525:root" }
```

You can specify a specific federated user ("Bob"):

```
"Principal": { "SGWS": "urn:sgws:identity::
27233906934684427525:federated-user/Bob" }
```

You can specify a specific federated group ("Managers"):

```
"Principal": { "SGWS": "urn:sgws:identity::
27233906934684427525:federated-group/Managers"  }
```

You can specify an anonymous principal:

```
"Principal": "*"
```

The Canonical User ID is not supported.

Specifying permissions in a policy

There are a set of permissions that you can specify in a policy. Each of these keywords maps to specific S3 Rest API operations.

Permissions applicable to buckets:

Permissions	S3 Rest API operations
s3:CreateBucket	PUT Bucket
s3:DeleteBucket	DELETE Bucket
s3:DeleteBucketPolicy	DELETE Bucket policy
s3:GetBucketAcl	GET Bucket ACL
s3:GetBucketConsistency	GET Bucket Consistency
s3:GetBucketLastAccessTime	GET Bucket Last Access Time
s3:GetBucketPolicy	GET Bucket policy
s3:GetBucketVersioning	GET Bucket versioning
s3:ListAllMyBuckets	GET Service, GET Storage Usage
s3:ListBucket	GET Bucket (List Objects)
s3:ListBucketMultipartUploads	List Multipart Uploads
s3:ListBucketVersions	GET Bucket versions
s3:PutBucketConsistency	PUT Bucket Consistency
s3:PutBucketLastAccessTime	PUT Bucket Last Access Time
s3:PutBucketPolicy	PUT Bucket policy

Permissions	S3 Rest API operations
s3:PutBucketVersioning	PUT Bucket versioning

Permissions applicable to objects:

Permissions	S3 Rest API operations
s3:AbortMultipartUpload	Abort Multipart Upload
s3:DeleteObject	DELETE Object
s3:DeleteObjectVersion	DELETE Object (a specific version of the object)
s3:GetObject	GET Object
s3:GetObjectAcl	GET Object ACL
s3:GetObjectVersion	GET Object, HEAD Object
s3:ListMultipartUploadParts	List Parts
s3:PutObject	PUT Object

Policies requiring special handling

Sometimes a policy can grant permissions that are dangerous for security, or dangerous for continued operations. For example, locking out the root user of the account. The StorageGRID Webscale S3 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

Policy description	Policy type	Amazon behavior	StorageGRID behavior
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

How client applications use certificates for security with REST APIs

When a client application establishes a TLS session to the StorageGRID Webscale system, the system sends a server certificate to the client application for verification to ensure that the HTTPS connection is secure.

The client application loads the grid CA certificate and uses it to verify that the client application is communicating with the expected StorageGRID Webscale system. This process protects against man-in-the-middle and impersonation attacks.

Supported hashing and encryption algorithms for TLS libraries

Client applications use the HTTPS protocol to communicate with the StorageGRID Webscale system over a network connection that uses Transport Layer Security (TLS). The StorageGRID Webscale system supports a limited set of hashing and encryption algorithms from the TLS libraries that client applications can use when establishing a TLS session. When you are setting up the communication processes, it is important for you to know which security algorithms the system uses.

The StorageGRID Webscale system supports the following cipher suite security algorithms:

- AES128-SHA
- AES256-SHA
- AES128-GCM
- AES256-GCM

AES128-SHA and AES256-SHA provide secure encryption and efficient processing of objects. AES128-GCM and AES256-GCM provide secure encryption and more efficient processing of large objects. The TLS session negotiates the connection, using either AES128 or AES256 based on the client application requirements, and the need to balance performance with encryption security.

Monitoring and auditing operations

You can monitor the health of your client application connections to the StorageGRID Webscale system by viewing summary attributes that list transaction counts for the LDR services on all Storage Nodes, or you can view the transactions for a specific Storage Node. Also, you can use audit messages to monitor the operations and transactions of the StorageGRID Webscale system.

Viewing transactions for S3 objects

You can view the number of successful and failed attempts by client applications to read, write, and modify S3 objects in the StorageGRID Webscale system. You can view a summary of all transactions for all LDR services, or you can view the transactions for a specific LDR service. You might want to do this to evaluate the health of the system.

Steps

1. Sign in to the Grid Management Interface using a supported browser.
2. Select **Grid**.
3. Select ***site*** > **Overview** > **Main**, and then view the **API Operations** area.

   The API Operations area displays a summary of information from all of the LDR services that support S3 client applications.
4. Select ***Storage Node*** > **LDR** > **S3** > **Overview** > **Main** to view information for individual LDR services.

Accessing and reviewing audit logs

The StorageGRID Webscale system securely and reliably transports audit messages from each service within the StorageGRID Webscale system to one or more audit repositories. API-specific (S3, Swift, and CDMI) audit messages provide critical security, operations, and performance monitoring data that can help you evaluate the health of your system.

About this task

The StorageGRID Webscale system compresses audit logs after one day and renames them using the format *YYYY-MM-DD.txt.gz* (where the original date is preserved).

Steps

1. Log in to the server using the user name and password as recorded in the `Passwords.txt` file.
2. Access the audit log directory through a command line of the server that hosts the AMS service.
3. Go to the `/var/local/audit/export/` directory.
4. View the `audit.log` file.

Related information

StorageGRID Webscale 10.3 Audit Message Reference

Benefits of active, idle, and concurrent HTTP connections

How you configure HTTP connections can impact the performance of the StorageGRID Webscale system. Configurations differ depending on whether the HTTP connection is active or idle or you have concurrent multiple connections.

Benefits of different types of HTTP connections

The type of HTTP connection duration can impact the performance of the StorageGRID Webscale system.

You can identify the performance benefits for the following types of HTTP connections:

- Idle HTTP connections
- Active HTTP connections
- Concurrent HTTP connections

Benefits of keeping idle HTTP connections open

You should keep HTTP connections open even when client applications are idle to allow client applications to perform subsequent transactions over the open connection. Based on system measurements and integration experience, you should keep an HTTP connection open for a maximum of 10 minutes. The LDR service might automatically close an HTTP connection that is kept open and idle for longer than 10 minutes.

Open and idle HTTP connections provide the following benefits:

- Reduced latency from the time that the StorageGRID Webscale system determines it has to perform an HTTP transaction to the time that the StorageGRID Webscale system can perform the transaction

  Reduced latency is the main advantage, especially for the amount of time required to establish TCP/IP and TLS connections.
- Increased data transfer rate by priming the TCP/IP slow-start algorithm with previously performed transfers
- Instantaneous notification of several classes of fault conditions that interrupt connectivity between the client application and the StorageGRID Webscale system

Determining how long to keep an idle connection open is a trade-off between the benefits of slow start that is associated with the existing connection and the ideal allocation of the connection to internal system resources.

Benefits of active HTTP connections

You should limit the duration of an active HTTP connection for a maximum of 10 minutes, even if the HTTP connection continuously performs transactions. Determining the maximum duration that a connection should be held open is a trade-off between the benefits of connection persistence and the ideal allocation of the connection to internal system resources.

Limited active HTTP connections provide the following benefits:

- Enables optimal load balancing across the StorageGRID Webscale system

To optimize load balancing across the StorageGRID Webscale system, you should prevent long-lived TCP/IP connections. You should configure client applications to track the duration of each HTTP connection and close the HTTP connection after a set time so that the HTTP connection can be reestablished and rebalanced.

The StorageGRID Webscale system balances its load when a client application establishes an HTTP connection. Over time, an HTTP connection that the StorageGRID Webscale system uses for a compute resource might no longer be optimal as load balancing requirements change. The system performs its best load balancing when client applications establish a separate HTTP connection for each transaction, but this negates the much more valuable gains associated with persistent connections.

- Allows maintenance procedures to start

  Some maintenance procedures start only after all the in-progress HTTP connections are complete.

- Allows client applications to direct HTTP transactions to LDR services that have available space.

Benefits of concurrent HTTP connections

You must keep multiple TCP/IP connections to the StorageGRID Webscale system open to allow idle connections to perform transactions as required. The number of client applications also affects how you handle multiple TCP/IP connections.

Concurrent HTTP connections provide the following benefits:

- Reduced latency

  Transactions can start immediately instead of waiting for other transactions to be completed.

- Increased throughput

  The StorageGRID Webscale system can perform parallel transactions and increase aggregate transaction throughput.

Client applications should establish multiple HTTP connections, either on a client-by-client basis or on a connection-pool basis. When a client application has to perform a transaction, it can select and immediately use any established connection that is not currently processing a transaction.

Each StorageGRID Webscale system's topology has different peak throughput for concurrent transactions and connections before performance begins to degrade. Peak throughput depends on factors such as computing resources, network resources, storage resources, and WAN links. The number of servers and services and the number of applications that the StorageGRID Webscale system supports are also factors.

StorageGRID Webscale systems often support multiple client applications. You should keep this in mind when you determine the maximum number of concurrent connections used by a client application. If the client application consists of multiple software entities that each establish connections to the StorageGRID Webscale system, you should add up all the connections across the entities. You might have to adjust the maximum number of concurrent connections in the following situations:

- The StorageGRID Webscale system's topology affects the maximum number of concurrent transactions and connections that the system can support.

- Client applications that interact with the StorageGRID Webscale system over a network with limited bandwidth might have to reduce the degree of concurrency to ensure that individual transactions are completed in a reasonable time.

- When many client applications share the StorageGRID Webscale system, you might have to reduce the degree of concurrency to avoid exceeding the limits of the system.

Separation of HTTP connection pools for read and write operations

You can use separate pools of HTTP connections for read and write operations and control how much of a pool to use for each. Separate pools of HTTP connections enable you to better control transactions and balance loads.

Client applications can create loads that are retrieve-dominant (read) or store-dominant (write). With separate pools of HTTP connections for read and write transactions, you can adjust how much of each pool to dedicate for read or write transactions.

Copyright information

Trademark information

How to send comments about documentation and receive update notifications

You can help us to improve the quality of our documentation by sending us your feedback. You can receive automatic notification when production-level (GA/FCS) documentation is initially released or important changes are made to existing production-level documents.

If you have suggestions for improving this document, send us your comments by email.

doccomments@netapp.com

To help us direct your comments to the correct division, include in the subject line the product name, version, and operating system.

If you want to be notified automatically when production-level documentation is released or important changes are made to existing production-level documents, follow Twitter account @NetAppDoc.

You can also contact us in the following ways:

- NetApp, Inc., 495 East Java Drive, Sunnyvale, CA 94089 U.S.
- Telephone: +1 (408) 822-6000
- Fax: +1 (408) 822-4501
- Support telephone: +1 (888) 463-8277

Index

F

G

H

I

L

M

O

P

Q

R

S

T

V

X