# StorageGRID® 9.0
# **CDMI Reference**

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) 4-NETAPP
Documentation comments: doccomments@netapp.com
Information Web: www.netapp.com

Part number 215-06987_A0
July 19, 2012

# Contents

# How the StorageGRID system implements CDMI

You can use a Cloud Data Management Interface (CDMI) client to connect to the CLB service or the LDR service in the StorageGRID system, and store and retrieve objects. The StorageGRID system uses its information lifecycle management (ILM) rules to manage objects in the grid.

## CDMI specification sections supported by the StorageGRID system

The StorageGRID system supports a number of sections from the Cloud Data Management Interface (CDMI) specification version 1.0.1 published by the Storage Networking Industry Association (SNIA).

### Data object resource operations

You must retrieve data objects by CDMI Object ID when using CDMI clients with the StorageGRID system.

**Note:** Objects ingested through SGAPI can be accessed through CDMI. Likewise, objects ingested through CDMI can be accessed through SGAPI. To access through CDMI, an object that was ingested through SGAPI, you must convert the object's UUID (returned in an SGAPI ingest response) to a CDMI Object ID.

| Section | Title |
|---|---|
| 8.4 | Read a data object (CDMI Content Type) |
| 8.5 | Read a data object (Non-CDMI Content Type) |
| 8.6 | Update a data object (CDMI Content Type; limited to wholesale metadata update) |
| 8.8 | Delete a data object (CDMI Content Type) |
| 8.9 | Delete a data object (Non-CDMI Content Type) |

### Byte range read operations

The StorageGRID system supports byte range read operations using both CDMI and Non-CDMI Content Types. For a Non-CDMI Content Type, the following byte range read operations are returned:

- If a single contiguous byte range is requested, the StorageGRID system returns the byte range.
- If multiple byte ranges are requested that can be coalesced without holes, the StorageGRID system returns a single coalesced range.

- If multiple byte ranges are requested that cannot be coalesced without holes, the StorageGRID system returns the entire object bytes.

If you enable compression (by using the **Grid Management > Grid Configuration > Configuration > Stored Object Compression** option) or if the object is retrieved from tape, the StorageGRID system locates and returns the requested portion of the object by reading the object starting at the beginning of the segment containing the first byte of the requested range. When compression is disabled and the object is retrieved from disk, the StorageGRID system is able to begin reading the segment from the start of the requested byte range and not the beginning of the segment. Thus, if compression is enabled or the object is retrieved from tape, it takes the StorageGRID system longer to return the requested portion of an object.

### Container object resource operations

The StorageGRID system does not support container objects. As a result, the StorageGRID system does not support named objects because named objects require container objects. However, the StorageGRID system does support some functions from the Container Object Resource Operations section of the CDMI specification.

**Note:** You must use the HTTP POST method to store nameless objects in the StorageGRID domain. Nameless objects have an object ID, but not a name. CDMI clients use a database to track the object IDs for nameless objects, and CDMI clients use object IDs to retrieve objects from the StorageGRID system.

| Section | Title |
|---|---|
| 9.9 | Create (POST) a new data object (CDMI Content Type) |
| 9.10 | Create (POST) a new data object (Non-CDMI Content Type) |

### Domain object resource operations

The StorageGRID system supports one hard-coded domain with a root URI of https://*IP_address*:*port*/CDMI/. Where *IP_address* is the IP address for the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service. Where *port* is the port number for the LDR service or the CLB service.

### Capability object resource operations

| Section | Title | Notes |
|---|---|---|
| 12.2 | Read a capabilities object (CDMI Content Type) | The StorageGRID system supports the following:<br>• System-wide capabilities<br>• Storage system metadata capabilities<br>• Data system metadata capabilities<br>• Data object capabilities |

### Metadata

| Section | Title | Notes |
|---|---|---|
| 16.3 | Storage system metadata | The StorageGRID system supports the following:<br>• cdmi_size<br>• cdmi_ctime<br>• cdmi_atime<br>• cdmi_hash<br>• cdmi_value_hash_provided |
| 16.4 | Data system metadata | The StorageGRID system supports:<br>• cdmi_data_redundancy<br>• cdmi_immediate_redundancy<br>• cdmi_value_hash |

The StorageGRID system converts the following existing StorageGRID metadata to populate the values of some CDMI storage system metadata. The following table identifies which StorageGRID metadata is used to populate CDMI system metadata. For more information, see the *StorageGRID Administrator Guide*.

| CDMI storage system metadata | StorageGRID metadata |
|---|---|
| cdmi_size | CSIZ |
| cdmi_ctime | CTME |
| cdmi_atime | LATM<br>The StorageGRID system uses the value from cdmi_ctime when an object lacks LATM (last access time) metadata. |
| cdmi_hash | The StorageGRID system returns the hash for the object. |
| cdmi_value_hash_provided | The StorageGRID system returns the name of the hash algorithm selected in the NMS MI when the grid stored the object. |

Objects stored in the StorageGRID system by StorageGRID API clients or by the FSG service can include non-CDMI metadata. For example, when StorageGRID API clients store objects in the StorageGRID system, the clients can use predefined metadata, which is a type of metadata that is designed specifically for StorageGRID API clients and the StorageGRID system. When you use CDMI clients to retrieve the objects, the response includes the predefined metadata.

### Related concepts

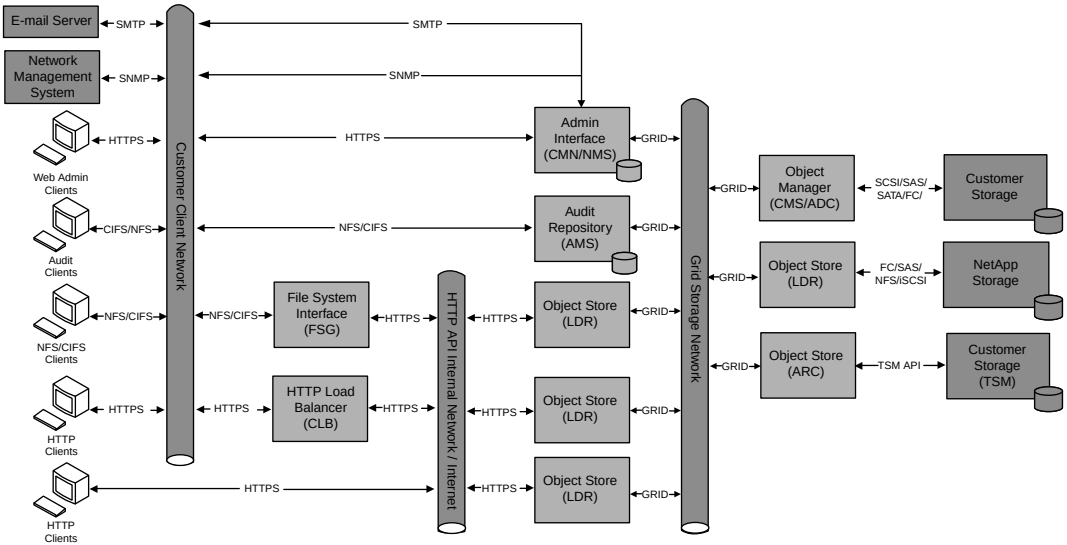
*How CDMI clients retrieve objects stored by StorageGRID API clients* on page 32

### Related tasks

*Retrieving CDMI capabilities with curl* on page 27

# What the StorageGRID system is

The StorageGRID system stores, protects, and preserves fixed-content data over its lifetime.




Different types of clients can submit content to the StorageGRID system for storage. Clients, such as NFS or CIFS clients, use the customer network to submit content to the FSG service. Other clients, such as HTTP clients, can use HTTPS connections to submit content directly to an LDR service, or HTTP clients can use the customer network to submit content directly to a CLB service. The StorageGRID system stores the submitted content as objects on different types of storage. Information lifecycle management (ILM) business rules instruct the StorageGRID system where to store the objects and how to manage the objects and their metadata over the lifetime of the object. Clients can retrieve objects at any time.

The StorageGRID system optionally supports gzip compression for both storage and retrieval. For more information about gzip compression, see section 14.1 of IETF RFC 2616.

**Note:** The StorageGRID system considers CDMI clients a type of HTTP client.

### Related information

*http://www.ietf.org*

## Grid nodes and services in the StorageGRID system

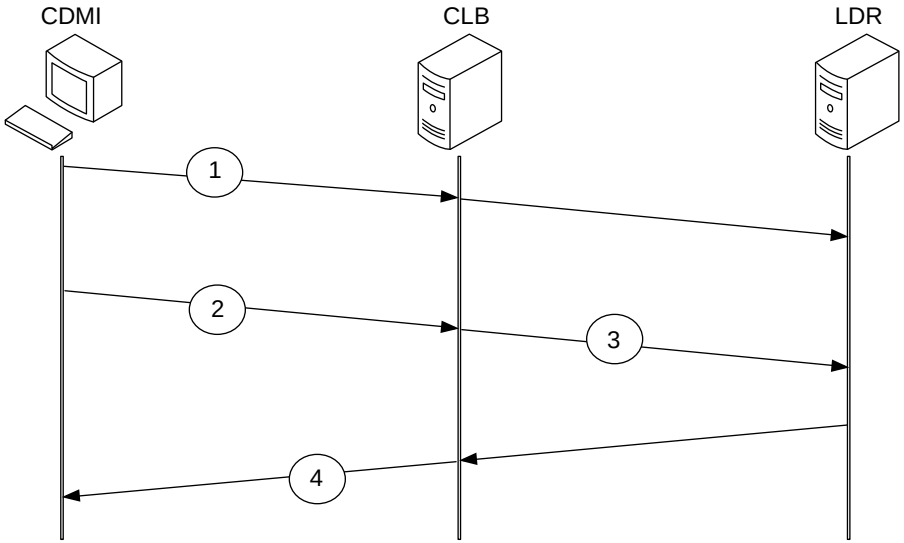
A deployment of the StorageGRID system consists of a collection of grid nodes and services that run on multiple virtual machines or servers to perform a specialized set of tasks.

| Service | Full name of the service | Description | Grid node to which the service belongs |
|---|---|---|---|
| ADC | Administrative Domain Controller | Maintains topology information and provides authentication services | Control Node |
| AMS | Audit Management System | Tracks grid activity and events | Admin Node or Audit Node |
| ARC | Archive | Communicates with archiving middleware to store and retrieve data to and from archive media | Archive Node |
| CLB | Connection Load Balancer | Acts as switchboard for connecting remote entities to the most efficient LDR<br>Primary connection point for remote entities using the HTTP protocols. | Gateway Node |
| CMN | Configuration Management Node | Manages grid-wide configurations, for example, connection profiles, grid tasks, and grid options | Primary Admin Node |
| CMS | Content Management System | Keeps track of the data stored in the grid<br>Stores content metadata and manages content replication based on ILM rules | Control Node |
| FSG | File System Gateway | Allows connections to the grid through standard file-sharing protocols (CIFS and NFS) | Gateway Node |
| LDR | Local Distribution Router | Stores, moves, verifies, and retrieves object data stored on disks | Storage Node |
| NMS | Network Management System | Monitors grid status and configures the grid | Admin Node |

| Service | Full name of the service | Description | Grid node to which the service belongs |
|---|---|---|---|
| SSM | Server Status Monitor | Monitors hardware performance, such as key operating system metrics and network metrics | Present on all grid nodes |

# How CDMI clients store objects (CLB service)

CDMI clients can use HTTP to connect directly with a CLB service and store objects. The CLB service identifies the optimal LDR service to satisfy client requests and forwards requests to the LDR service.



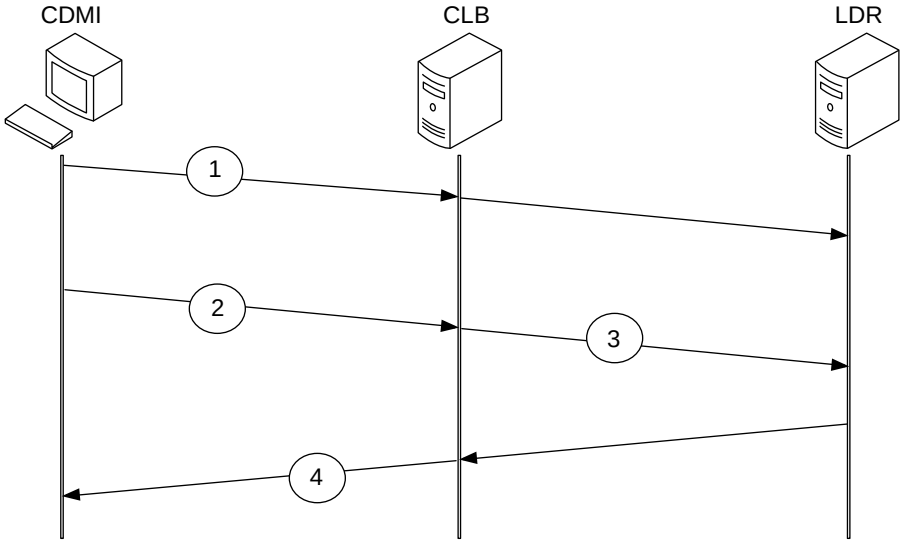

1. The CDMI client opens an HTTPS connection to the configured HTTP port for a CLB service. The CLB service acts as a proxy for the LDR services.
2. The CDMI client issues an HTTP POST request that includes the object and any metadata.
3. The CLB service identifies the optimal LDR service to satisfy client requests and forwards requests to the LDR service.

   **Note:** The CLB service uses ranking criteria to identify which LDR service to use. As a result, the CDMI client does not have to identify which LDR service to use.

4. After the LDR service stores a copy of the object, the LDR returns the object ID to the CDMI client through the CLB service.

# How CDMI clients retrieve objects (CLB service)

CDMI clients can use HTTP to connect directly with a CLB service and retrieve objects. The CLB service identifies the optimal LDR service to satisfy client requests and forwards requests to the LDR service.




1. The CDMI client opens an HTTPS connection to the configured HTTP port for a CLB service. The CLB service acts as a proxy for the LDR services.
2. The CDMI client issues an HTTP GET request that includes the object ID for the object that it wants to retrieve.

   **Note:** CDMI clients must use the object ID to retrieve objects from the StorageGRID system.
3. The CLB service identifies the optimal LDR service to satisfy the request and forwards the request to the LDR service.

   **Note:** The CLB service uses ranking criteria to identify which LDR service to use. As a result, the CDMI client does not have to identify which LDR service to use.
4. The LDR service returns the object and any requested metadata.

# How CDMI clients retrieve objects (LDR service)

CDMI clients can use HTTP to connect directly with an LDR service and retrieve objects.

1. The CDMI client opens an HTTPS connection to the configured HTTP port for an LDR service.
2. The CDMI client issues an HTTP GET request that includes the object ID.

3. The LDR service returns the object and any requested metadata.

A CDMI client can connect directly with multiple LDR services. A connection with one or more LDR services enables high-performance parallel transfers and eliminates the single point of failure that is associated with connecting to a CLB service.

# How ILM manages CDMI objects in the StorageGRID system

Information lifecycle management (ILM) in the StorageGRID system enables you to use metadata in rules to automatically manage CDMI objects in the grid.

You can use *CDMI protocol handler version*, *last access time*, and *CDMI user-defined* metadata in ILM rules for objects stored in the grid by CDMI clients. You can use *CDMI protocol handler version* metadata to identify all the objects stored by a CDMI client and to perform specific actions on those objects. For example, you can specify where to store CDMI objects and for how long. You can use *last access time* metadata to identify content that has not been retrieved in two years and move the content to a cheaper grade of storage. In addition, you can set the filter criteria to evaluate objects against *CDMI user defined* metadata. The CDMI protocol handler version and last access time are StorageGRID metadata. For information about setting up ILM rules, see the ILM chapter in the *StorageGRID Administrator Guide*.

# Supported hash algorithms for objects

The StorageGRID system can use either SHA-1 or SHA-2 256 hash algorithms to generate a hash for each object stored in the grid, and you can choose which algorithm to use.

In the NMS MI, you can use the **Grid Management** > **Grid Configuration** > **Configuration** > **Main** > **Stored Object Hashing** option to change the hash algorithm for the StorageGRID system. Because you can change the algorithm, you might have objects in the grid with hashes generated by different algorithms. As a result, metadata for different objects might include different algorithm names. The algorithm name associated with the object depends on which algorithm was selected in the NMS MI when the object was stored in the grid.

The following table maps the choices in the NMS MI to the names of the hash algorithms.

| Algorithm choice in NMS MI | Name of algorithm |
|---|---|
| SHA-256 | SHA-2 256 bits |
| SHA-1 | SHA-1 |

# How the StorageGRID system implements immediate redundancy

The StorageGRID system supports the CDMI Data System Metadata Capabilities functionality `cdmi_data_redundancy` and `cdmi_immediate_redundancy` to save up to two copies of an

object to two Storage Nodes at object creation. This functionality provides protection against data loss should a Storage Node fail.

A CDMI client specifies redundancy with the Data System Metadata `cdmi_data_redundancy` and `cdmi_immediate_redundancy` in a CDMI `POST` request when creating an object. If this metadata is not included in a CDMI Content Type request, the following defaults are assumed:

- `"cdmi_data_redundancy" : "2"`
- `"cdmi_immediate_redundancy" : true`

These defaults also apply to Non-CDMI Content Type `POST` object create requests. For CDMI Content Type requests, you can display redundancy by setting `cdmi_data_redundancy` to `false`.

Internally, the StorageGRID system achieves redundancy by using Dual Commit, which creates two copies of an object. Thus, a `cdmi_data_redundancy` request with a value greater than `2` creates two copies and not the requested value.

> **Note:** Dual Commit creates two copies of an object before ILM rules are evaluated, which might create additional copies. For more information about Dual Commit and ILM rules, see the *StorageGRID Administrator Guide*.

The following table summarizes the responses for successful redundancy requests:

| **CDMI client request** | | **Does the grid use Dual Commit?** | **CDMI response for success** | |
|---|---|---|---|---|
| **cdmi_data_redundancy** | **cdmi_immediate_redundancy** | | **cdmi_data_redundancy_provided** | **cdmi_immediate_redundancy_provided** |
| not present | not present | Yes | 2 | true |
| 0 | true | No | 1 | true |
| 1 | true | No | 1 | true |
| 2 | true | Yes | 2 | true |
| not present | false | No | 1 | true because one object was stored |
| a number greater than 2 | true | Yes | 2 | true<br>**Note:** The grid stores a maximum of 2 copies. |

The following table summarizes the responses for failed redundancy requests:

| **CDMI client request** | | **Does the grid use Dual Commit?** | **CDMI response for failure** | |
|---|---|---|---|---|
| **cdmi_data_redundancy** | **cdmi_immediate_redundancy** | | **cdmi_data_redundancy_provided** | **cdmi_immediate_redundancy_provided** |
| not present | not present | No | 1 | false |
| 0 | true | No | 1 | true |
| 1 | true | No | 1 | true |
| 2 | true | No | 1 | false |
| not present | false | No | 1 | true |
| a number greater than 2 | true | No | 1 | false |

After the grid creates a copy for the object, it evaluates the ILM rules for the copy and the object. The rules in the ILM policy determine the actions the StorageGRID system takes with the copy and the object. For example, the ILM rules might instruct the grid to make additional copies of the object in different locations and delete the original copy. For more information about Dual Commit and ILM rules, see the *StorageGRID Administrator Guide*.

# CDMI namespace permissions you can specify in the StorageGRID system

You can specify permissions for CDMI clients in the CDMI namespace in the StorageGRID system.

## CDMI client access types you can specify

In the NMS MI, you can specify whether a CDMI client can read, write, modify, delete, or query data objects in the CDMI namespace. You can also specify whether to enable last access time metadata.

You can grant or deny the following types of access to CDMI clients in the NMS Management Interface (MI) in the StorageGRID system:

- **Read**
- **Modify/Write**
- **Delete**
- **Query**
- **Last Access Time**

The following table identifies the HTTP method used for each type of access.

| Access type | HTTP method |
|---|---|
| **Read** | `GET` |
| **Modify/Write** | `PUT` for the modify access type<br>`POST` for the write access type |
| **Delete** | `DELETE` |
| **Query** | The StorageGRID system does not support the **Query** option for CDMI clients. |
| **Last Access Time** | `GET`<br>You must enable both **Last Access Time** and **Read**. When a CDMI client uses `GET` to retrieve an object, the grid stores the time that the CDMI client retrieved the object in internal object metadata called last access time metadata. |

# Read access to objects

The **Read** access type determines whether a CDMI client has permission to retrieve objects and object metadata from the CDMI namespace.

**Note:** For objects stored in the grid by StorageGRID API clients, you must derive the CDMI Object ID from the StorageGRID UUID before you can read or retrieve the object.

## Retrieve objects and object metadata (GET)

CDMI clients use the HTTP `GET` method and an object ID to read objects and object metadata in the CDMI namespace.

## Retrieve object metadata (GET)

CDMI clients use the HTTP `GET` method and an object ID to retrieve object metadata from the CDMI namespace.

The response might include predefined metadata or custom metadata when a StorageGRID API client created the object in the StorageGRID system.

## Predefined metadata

Only StorageGRID API clients can use predefined metadata with the StorageGRID system.

Unless otherwise noted, predefined metadata becomes read-only after the grid ingests it, and you cannot delete it. Predefined metadata use the X-BYC-*XXXX* format where *XXXX* is one of the following:

| Metadata | Set by | Status in grid | Description |
|---|---|---|---|
| XAID | FSG | Read-only | Identifies the StorageGRID API client that stored the object<br>A NetApp Solutions Engineer provides the value for the StorageGRID API client to use. |
| XTYP | FSG | Read-only | Identifies the type of object saved to the grid |
| XVER | Storage GRID API client or FSG | Read-only | Indicates the version of the metadata<br>Defined by the StorageGRID API client |

| **Metadata** | **Set by** | **Status in grid** | **Description** |
|---|---|---|---|
| MCLS | Storage GRID API client | Read-only | Indicates the TSM management class<br>Defined by the StorageGRID API client |
| STR0-STR9 | Storage GRID API client | Read-write<br>Can be deleted | Identifies a string value<br>Value defined by the StorageGRID API client |
| NUM0-NUM9 | Storage GRID API client | Read-write<br>Can be deleted | Identifies a numerical value<br>Value defined by the StorageGRID API client |
| FPTH | FSG | Read-only | Indicates the FSG file path at ingest |
| MODE | FSG | Read-only | Indicates the file system status for the object at ingest |
| FUID | FSG | Read-only | Indicates the user ID associated with the object at ingest |
| FGID | FSG | Read-only | Indicates the group ID associated with the object at ingest |
| CTIM | FSG | Read-only | Indicates the Linux ctime value associated with the object at ingest |
| MTIM | FSG | Read-only | Indicates the modification time associated with the object at ingest |
| FGRP | FSG | Read-only | Indicates the FSG replication group of the FSG that ingested the object |
| FSGN | FSG | Read-only | Indicates the FSG backup node ID for the object |
| RPLG | FSG | Read-only | Indicates the FSG replication group for the FSG backup |
| NSID | FSG | Read-only | Indicates the next FSG backup pending session ID |
| NXSC | FSG | Read-only | Indicates the sequence count of the next FSG replication message to be processed in the next session pending |
| NSNI | FSG | Read-only | Indicates the node ID of the next session pending |
| BUID | FSG | Read-only | Indicates the FSG backup ID |

# Modify or write access

The **Modify/Write** access type determines whether a client has permission to store objects and update object metadata in the namespace.

## Store objects (POST)

CDMI clients use the HTTP `POST` method to store objects in the CDMI namespace.

When you use CDMI clients to store content in the StorageGRID system, you can only store nameless objects; you cannot store named objects.

> **Note:** You cannot use the `PUT` method to store objects in the StorageGRID system. The `PUT` method requires CDMI containers, and the StorageGRID system does not support CDMI containers.

By default, the StorageGRID system enables immediate redundancy for all `POST` requests for the CDMI content type and the non-CDMI content type. You can disable immediate redundancy for the CDMI content type, but not the non-CDMI content type. However, it is recommended that you include cdmi_immediate_redundancy metadata set to true in all the `POST` requests to enable immediate redundancy and protect against data loss.

#### Related concepts

*How the StorageGRID system implements immediate redundancy* on page 12

## Update object user metadata (PUT)

CDMI clients use the HTTP `PUT` method and an object ID to update object user metadata in the CDMI namespace.

The StorageGRID system only supports adding or updating all user metadata for an object and does not support adding or updating an individual user metadata (using the URI syntax `?metadata:<metadataname>`). Additionally, updating other fields, for example `value` or `mimetype`, using the `PUT` method, is not supported.

# Delete access

The **Delete** access type determines whether a CDMI client has permission to delete objects from the namespace.

## Query access

The **Query** access type determines whether a CDMI client has permission to perform queries in the namespace.

**Note:** The StorageGRID system does not support the **Query** option for CDMI clients.

## Last access time

The **Last Access Time** permission determines whether the grid updates last access time metadata for an object when a CDMI client retrieves the object. You can create Information Lifecycle Management (ILM) rules to take action on objects based on the last time that a CDMI client retrieved the object.

When a CDMI client that is assigned an HTTP profile with **Last Access Time** enabled uses `GET` to retrieve an object, the grid saves the retrieval time in internal object metadata called last access time metadata.

Only the grid can use internal metadata. For example, ILM policies can use last access time metadata to identify when an object was last retrieved. For more information about last access time metadata and ILM policies, see the *StorageGRID Administrator Guide*.

**Note:** Because last access time metadata updates each time that a CDMI client retrieves an object, it can affect grid performance. It is recommended that you disable **Last Access Time** in the NMS MI when no ILM policies use last access time metadata.

# Connecting clients to the StorageGRID system

You must configure the StorageGRID system to accept HTTP connections from CDMI clients. CDMI clients use the HTTP connections to access and communicate with the StorageGRID system.

**Note:** IPv6 is only supported for HTTP client connections through the CLB service. For more information about support for IPv6, see the *StorageGRID Administrator Guide*.

## Configuring client connections

A number of steps are required to configure the StorageGRID system to accept HTTP connections from CDMI clients.

### Steps

1. *Associating client IP addresses with link-cost groups* on page 20
   You can associate a link-cost group with the IP addresses that clients use to connect with the grid. The link-cost group allows the grid to route clients to the appropriate servers.
2. *Creating HTTP profiles of namespace permissions* on page 21
   HTTP profiles identify whether read, write, modify, query, or delete are enabled or disabled in a namespace. You can create multiple HTTP profiles.
3. *Associating HTTP profiles with client IP addresses* on page 22
   You can associate HTTP profiles with individual clients or with groups of clients, based on IP addresses. The association gives clients access to the StorageGRID namespace and identifies the HTTP permissions for the client in the namespace.

### Associating client IP addresses with link-cost groups

You can associate a link-cost group with the IP addresses that clients use to connect with the grid. The link-cost group allows the grid to route clients to the appropriate servers.

#### About this task

Servers for the StorageGRID system are organized into link-cost groups. Link-cost groups identify the cost of operating the group of servers. The grid can improve performance when you associate a link-cost group with clients. The grid uses the IP address and the link-cost group to route clients to the LDR service or CLB service on the appropriate servers.

#### Steps

1. In the NMS MI, go to **Grid Management > Grid Configuration > Link Cost Groups > Configuration > Main**.

2. In the **Client Group IP Ranges** table, perform one of the following actions:

| When... | Then... |
|---|---|
| No entries exist | Click **Edit**. |
| One or more entries exist | Click **Insert**. |

3. In the **IP Range Name** box, type a name for the IP address or the range of IP addresses.

   You can use any name. The grid configuration does not reference the name elsewhere.

4. In the **IP Range** box, type the IP address or the range of IP addresses that the client will use to contact the StorageGRID system.

   Use a hyphen or slash to indicate an inclusive range of IP addresses, for example:

   - 192.168.120.0/24 (CIDR format)
   - 192.168.142.20-192.168.142.28 (dotted decimal format)

   You can use an abbreviated format for masks in eight-bit steps. For example, 192.168.142.0 is equivalent to the CIDR notation 192.168.142.0/24, and you can extend it as follows: n.n.0.0 is equivalent to n.n.0.0/16.

5. In the **Group ID** list, select an ID.

   The ID number identifies the group of servers to which the client with the specified IP address should connect.

6. Click **Apply Changes**.

7. Repeat this procedure for each range of IP addresses that clients will use to access the StorageGRID system.

## Creating HTTP profiles of namespace permissions

HTTP profiles identify whether read, write, modify, query, or delete are enabled or disabled in a namespace. You can create multiple HTTP profiles.

### Steps

1. In the NMS MI, go to **Grid Management > HTTP Management > Permissions > Configuration > Main**.

2. In the **HTTP /CDMI and /UUID Namespace** table, perform one of the following actions:

| When... | Then... |
|---|---|
| No entries exist | Click **Edit**. |
| One or more entries exist | Click **Insert**. |

3. Select the check boxes for the HTTP operations that you want to enable in the profile.

The StorageGRID system does not support the **Query** operation in the **HTTP /CDMI and / UUID Namespace** for CDMI clients.

4. Click **Apply Changes**.
5. Create additional profiles as needed.

## Associating HTTP profiles with client IP addresses

You can associate HTTP profiles with individual clients or with groups of clients, based on IP addresses. The association gives clients access to the StorageGRID namespace and identifies the HTTP permissions for the client in the namespace.

### Steps

1. In the NMS MI, go to **Grid Management > HTTP Management > Clients > Configuration > Main**.
2. In the **HTTP Entities** table, perform one of the following actions:

| When... | Then... |
|---|---|
| No HTTP entities exist | Click **Edit**. |
| One or more HTTP entities exist | Click **Insert**. |

3. In the **Description** box, type a description of the client.
4. In the **IP Range** box, type the range of IP addresses that the client can use to connect to the LDR service or the CLB service.
5. In the **Profile Name** list, select the name of the HTTP profile that you created.
6. Click **Apply Changes**.

# How client authentication works

The StorageGRID system uses its HTTP management settings to authenticate client requests for access to the grid.

When a CDMI client requests access to the grid, the StorageGRID system authenticates the request against the HTTP management settings that you created for the CDMI client.

1. The StorageGRID system checks that the CDMI client is using the same IP address or range of IP addresses that are defined in the HTTP management settings.
2. When the client passes the authentication process, the StorageGRID system opens a TCP/IP connection.

# Copying the CA certificate from the StorageGRID system

You can copy the certificate authority (CA) certificate from the NMS MI in the StorageGRID system for clients that require server verification.

### Steps

1. In the NMS MI, go to **Grid Management > Grid Configuration > Overview > Main**.
2. Under **Grid Information**, expand **CA Certificate**.
3. Select the CA certificate.

   Include the BEGIN CERTIFICATE header and the END CERTIFICATE footer in your selection.
4. Right-click the selected certificate, and then select **Copy**.

# Testing client connections to the StorageGRID system

You can test the HTTP connection between the client and the StorageGRID system to ensure that the connection works. You can also test that the CDMI client can store objects in the StorageGRID system and retrieve objects. Multiple testing methods are provided in case one method does not work for you.

If you copy a command from this section and paste the command into another application, the copy-and-paste process might remove dashes that appear between words near a line break. You must ensure that the pasted command includes all dashes before you run the command.

## Finding IP addresses for grid nodes

You can find the IP address in the NMS MI for Storage Nodes that host the LDR service or Gateway Nodes that host the CLB service. You need the IP address to connect CDMI clients to the LDR service or the CLB service.

### Steps

1. In the NMS MI, expand **Grid Topology**.
2. In the Grid Topology tree, locate and expand the Storage Node or Gateway Node to which you want to connect.

   The name of the Storage Node or Gateway Node depends on the configuration of the grid.

   The services for the selected grid node appear.
3. Expand the **SSM** service, click **Resources**, and scroll to the **Network Address** table.

   Depending on your grid configuration, IP addresses appear for one or more of the following: **eth0**, **eth1**, **eth2**, and so on.
4. Perform one of the following actions:

| To connect CDMI clients to... | Choose this IP address... |
|---|---|
| API Gateway Node, Basic Gateway Node, or Storage Node | If the **Network Address** table only lists **eth0**, use the IP address for **eth0**.<br>If the **Network Address** table lists both **eth0** and **eth1** (and a dedicated NFS storage network does not exist), use the IP address for **eth1**. |

| To connect CDMI clients to... | Choose this IP address... |
|---|---|
| High Availability Gateway Node | If the **Network Address** table lists both **eth0** and **eth1**, use either of the IP addresses for **eth0** or **eth1**.<br>If the **Network Address** table lists **eth0**, **eth1**, and **eth2**, use the IP Address for **eth1**. |

You can establish HTTP connections from CDMI clients to any of the listed IP addresses. However, you typically want to use the IP address on the customer network instead of the IP address on the grid network or the heartbeat network.

### After you finish

Your next step is to locate the port number for the CLB service or the LDR service to which you want to connect the CDMI client.

### Related tasks

*Finding port numbers for the LDR service and the CLB service* on page 25

# Finding port numbers for the LDR service and the CLB service

You can find the port numbers for the LDR service and the CLB service in the NMS MI. You require the port numbers to create an HTTP connection from CDMI clients to the LDR service on Storage Nodes or the CLB service on Gateway Nodes.

### About this task

The grid might use the default port numbers or the custom port numbers. You should use the NMS MI to confirm which port numbers the grid uses.

### Steps

1. In the NMS MI, go to **Grid Management** > **Grid Configuration** > **Storage** > **Main**.
2. Scroll to the **Ports** table and locate the port numbers for the LDR and CLB services.

### Related concepts

*Default HTTP ports for the CLB and LDR services* on page 35

# Testing HTTP connections with telnet

You can use telnet to test the HTTP connection between CDMI clients and the StorageGRID system to ensure that the HTTP connection is correctly configured.

### Before you begin

- You must have configured an IP address for the CDMI client in the NMS MI.
- You must know the IP address and port number for the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service.

### About this task

You can connect the CDMI client to the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service.

### Step

1. From a CDMI client, use telnet to connect to the CLB service or the LDR service by entering the following command:

   **`telnet`** ***`IP_address port`***

   For *`IP_address`* and *`port`*, use the IP address and port for the Gateway Node that hosts the CLB Service or the Storage Node that hosts the LDR service.

   If you correctly configured the IP address for the CDMI client in the NMS MI, a delay of several seconds occurs, and then the CLB service or the LDR service drops the connection.

   If you incorrectly configured the IP address for the CDMI client in the NMS MI, the connection closes immediately. If the CLB service or the LDR service is not running or if a network error occurs, telnet is unable to connect to the CLB or the LDR service.

# Testing HTTP connections with openssl

You can use the `openssl` command to test the HTTP connection between CDMI clients and the StorageGRID system to ensure that the HTTP connection is correctly configured.

### Before you begin

- You must have configured an IP address for the CDMI client in the NMS MI.
- You must know the IP address and port number for the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service.

### About this task

You can connect the CDMI client to the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service.

### Step

1. From a CDMI client, establish an HTTP connection to the CLB service or the LDR service by entering the following command:

   **openssl s_client -tls1 -connect *IP_address*:*port***

   For *IP_address* and *port*, you must use the IP address and port for the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service.

   If you correctly configured the IP address for the CDMI client in the NMS MI, a connected response appears.

   If you incorrectly configured the IP address for the CDMI client in the NMS MI, an error response appears.

## Retrieving CDMI capabilities with curl

You can retrieve the CDMI capabilities of the StorageGRID system to see the CDMI functions that the StorageGRID system supports. Knowing the CDMI capabilities helps you understand the functions that CDMI clients can perform with the StorageGRID system.

### Before you begin

You must know the IP address and port number for the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service.

### About this task

In the following task, *IP_address* and *port* are for the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service.

### Steps

1. From a CDMI client, use curl to retrieve CDMI capabilities from the StorageGRID system by entering the following command:

   **curl -X GET --header 'Host: *IP_address*:*port*' 'Content-Type: application/cdmi-capability' --header 'X-CDMI-Specification-Version: 1.0.1' -k https://*IP_address*:*port*/CDMI/cdmi_capabilities/**

   A response that looks similar to the following appears:

   ```
   {"objectType":"application/cdmi-
   capability","objectID":"00006FFD0009B74801","objectName":"cdmi_capabilit
   ies/","parentURI":"/","parentID":"00006FFD0009744905","capabilities":
   ```

```
{"cdmi_domains":"true","cdmi_metadata_maxitems":"32","cdmi_metadata_maxs
ize":"4096","cdmi_metadata_maxtotalsize":"32768","cdmi_object_access_by_
ID":"true","cdmi_post_dataobject_by_ID":"true","cdmi_security_data_integ
rity":"true"},"childrenrange":"0-1","children":
["dataobject/","domain/"]}
```

The supported CDMI capabilities are displayed after `"capabilities"`, for example, `"cdmi_domains":"true"`.

*`IP_address`* and *`port`* are for the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service.

2. Optional: Retrieve CDMI capabilities for other types of CDMI objects, such as domain and data objects, by appending the required capability name to the URL with a trailing forward slash.

   For more information about the different objects, see the CDMI specification.

   **Example**
   The following command retrieves the data system metadata capabilities:

   **`curl -X GET --header 'Host: IP_address:port' 'Content-Type: application/cdmi-capability' --header 'X-CDMI-Specification-Version: 1.0.1' -k https://IP_address:port/CDMI/cdmi_capabilities/dataobject/`**

   *`IP_address`* and *`port`* are for the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service.

   A response similar to the following appears:

```
{"objectType":"application/cdmi-
capability","objectID":"00006FFD0009B60802","objectName":"dataobject/","
parentURI":"/
cdmi_capabilities/","parentID":"00006FFD0009B74801","capabilities":
{"cdmi_read_value":"true","cdmi_read_value_range":"true","cdmi_read_meta
data":"true","cdmi_delete_dataobject":"true","cdmi_size":"true","cdmi_ct
ime":"true","cdmi_atime":"true","cdmi_data_redundancy":"2","cdmi_immedia
te_redundancy":"true","cdmi_hash":"true","cdmi_value_hash":
["SHA256"]},"childrenrange":"","children":[]}
```

   This response includes the "cdmi_value_hash" capability, indicating that the SHA-256 hash algorithm is supported.

### Related references

*CDMI specification sections supported by the StorageGRID system* on page 5

# Testing object storage and retrieval with curl

You can store and retrieve a test object to ensure that these functions work.

### Before you begin

You must know the IP address and port number for the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service.

### About this task

You can connect CDMI clients to the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service. In the following steps, *`IP_address`* and *`port`* are for the Gateway Node that hosts the CLB service or the Storage Node that hosts the LDR service.

### Steps

1. From a CDMI client, use curl to store a test file in the grid by entering the following command:

   **`curl -X POST --header 'Host: IP_address:port' 'Accept: application/cdmi-object' --header 'Content-Type: application/cdmi-object' --header 'X-CDMI-Specification-Version: 1.0.1' -d '{"domainURI":"/cdmi_domains/", "value":"Hello Big World"}' -k https://IP_address:port/CDMI/cdmi_objectid/`**

   A response similar to the following example appears:

   ```
   {"capabilitiesURI":"/cdmi_capabilities/
   dataobject/","completionStatus":"Complete","domainURI":"/
   cdmi_domains/","mimetype":"text/
   plain","objectID":"00006FFD0019692A00FAE83433343A47939555F14AA4D2F115","
   objectType":"application/cdmi-object","metadata":
   {"cdmi_atime":"2012-04-03T20:20:20.994783Z","cdmi_ctime":"2012-04-03T20:
   20:20.994783Z","cdmi_data_redundancy_provided":"2","cdmi_hash":"EBBA17EF
   9E791C8571BDD194FF874245B576EBCB44C8AAECA8DD8B08DECFCC8D","cdmi_immediat
   e_redundancy_provided":"true","cdmi_size":"15","cdmi_value_hash_provided
   ":"SHA256"}}
   ```

   The response includes a `"completionStatus":"Complete"` that indicates you successfully created the object. The response also includes the `"objectID"` number for the test object. In this example, the `"objectID"` is `00006FFD0019692A00FAE83433343A47939555F14AA4D2F115`.

2. Copy the `"objectID"` number from the response.

3. Use curl to retrieve the test object from the grid by entering the following command that includes the `"objectID"` number:

   **`curl -X GET --header 'Host: IP_address:port' 'Accept: application/cdmi-object' --header 'X-CDMI-Specification-Version: 1.0.1' -k https://`**

***IP_address*:*port*/CDMI/cdmi_objectid/**
**00006FFD0019692A00FAE83433343A47939555F14AA4D2F115**

A response similar to the following example appears:

```
{"capabilitiesURI":"/cdmi_capabilities/
dataobject/","completionStatus":"Complete","domainURI":"/
cdmi_domains/","mimetype":"text/
plain","objectID":"00006FFD0019692A00FAE83433343A47939555F14AA4D2F115","
objectType":"application/cdmi-
object","valuetransferencoding":"utf-8","metadata":
{"cdmi_atime":"2012-04-03T20:20:20.994783Z","cdmi_ctime":"2012-04-03T20:
20:20.994783Z","cdmi_hash":"EBBA17EF9E791C8571BDD194FF874245B576EBCB44C8
AAECA8DD8B08DECFCC8D","cdmi_size":"15","cdmi_value_hash_provided":"SHA25
6"},"valuerange":"0-14","value":"Hello Big World"}
```

4. Optional: Retrieve the value of a specific field by replacing `/CDMI/cdmi_objectid/00006FFD0019692A00FAE83433343A47939555F14AA4D2F115` with `/CDMI/cdmi_objectid/00006FFD0019692A00FAE83433343A47939555F14AA4D2F115?*field_name*`.

   The response includes the value for the requested field for the object.

5. Optional: Retrieve all metadata that begins with the prefix `cdmi` by replacing `/CDMI/cdmi_objectid/00006FFD0019692A00FAE83433343A47939555F14AA4D2F115` with `/CDMI/cdmi_objectid/00006FFD0019692A00FAE83433343A47939555F14AA4D2F115?metadata:*cdmi_*`.

   The response includes all metadata for the object that begins with the `cdmi` prefix.

# Using CDMI clients with the StorageGRID system

You can use the StorageGRID system to manage CDMI client access to the grid by changing the state of HTTP connections to the grid. You can also use the NMS MI to view HTTP transactions for CDMI clients and to look up object IDs. In addition, you can use CDMI clients to retrieve content stored by StorageGRID API clients.

## Managing HTTP connections

In the NMS MI, you can change the state of an HTTP connection to the grid to online, online (read-only), redirect, or offline to manage client access to the grid.

### About this task

The state of the HTTP connection affects StorageGRID API clients and CDMI clients. For example, when you change **HTTP State** to **Offline**, neither StorageGRID API clients nor CDMI clients can access the grid because the HTTP connection is closed.

### Steps

1. Go to **Grid Topology > LDR > Configuration > Main**.
2. In the **HTTP/CDMI State** list, select a state.

## Viewing HTTP transactions for CDMI

You can view the number of successful and failed attempts by CDMI clients to read, write, and modify objects in the StorageGRID system. You can view a summary of all transactions for all LDR services, or you can view the transactions for a specific LDR service.

### Steps

1. In the NMS MI, go to **Grid > Overview > Main**, and view the **API Operations** area.

   The **API Operations** area displays a summary of information from all of the LDR services in the grid that support CDMI clients.

2. You can view information for individual LDR services by going to **Grid Topology > LDR > CDMI > Overview > Main**.

   You can reset the counter for the LDR service to zero. On the **Configuration** page, select the **Reset CDMI Counts** check box, and click **Apply Changes**. The numbers on the **Main** page reset to zero and start counting up again.

# Viewing information about objects

You can type or paste an object ID in the NMS MI to view information about the object in the grid.

### Steps

1. Obtain the object ID from the CDMI client.
2. In the NMS MI, go to **Grid Topology > CMN > Object Lookup**.
3. Click **Configuration**.
4. In the **Object Identifier** box, type or paste the object ID, and click **Apply Changes**.

   **Note:** If you paste or type an invalid object ID, an error message appears.
5. Click **Overview** to review the results.

# How CDMI clients retrieve objects stored by StorageGRID API clients

You can use CDMI clients to retrieve objects stored by StorageGRID API clients after you derive an object ID from the UUID for the stored objects. Understanding how the StorageGRID system uses UUIDs and how CDMI uses object IDs helps you derive an object ID from a UUID.

## How the StorageGRID system uses UUIDs

The StorageGRID system uses the LDR service to assign a universally unique identifier (UUID) to each object in the grid.

UUIDs are 128 bits with an internal binary structure and a string representation in the form of A-B-C-D-E:

- A is 8 hex digits.
- B is 4 hex digits.
- C is 4 hex digits.
- D is 4 hex digits.
- E is 12 hex digits.

Each hex digit can take the values from 0 to 9, A through F (or lowercase a through lowercase f). Following is an example of a UUID: F81D4fAE-7DEC-11D0-A765-00A0C91E6BF6.

The grid randomly generates the UUID for each object, as described in section 4.4 of IETF RFC 4122.

**Related information**

*http://www.ietf.org*

## How the StorageGRID system uses UUIDs and CDMI object IDs

The StorageGRID system uses universally unique identifiers (UUIDs) to track and manage content, but CDMI clients use object IDs to track objects.

When a Gateway Node or a StorageGRID API client submits content to the StorageGRID system, the StorageGRID system stores the content and treats the stored content as one or more objects. The LDR service assigns a UUID to each object, and the StorageGRID system uses the UUID to track and manage the object.

When a CDMI client submits content to the StorageGRID system, the StorageGRID system stores the content and treats the stored content as one or more objects. However, the LDR service creates a UUID, embeds the UUID in an object ID, and assigns the object ID to each object that CDMI clients submit to the grid. CDMI clients use the OID to retrieve, update, and delete objects.

## Ruby code examples for deriving an object ID from a UUID

Ruby code examples show how you can derive a CDMI object ID from a StorageGRID UUID. You might have to derive CDMI object IDs if you want to use a CDMI client to retrieve the content stored in the grid by a StorageGRID API client or a Gateway Node.

CDMI clients require the object ID for an object to retrieve, update, or delete the object.

You must be familiar with section 5.11 of the CDMI specification that defines the form of CDMI object IDs to understand the Ruby code examples in this topic. The code examples use the following information:

- Enterprise number for the StorageGRID system is 28669 (0x006FFD).
- Length of the object ID for a data object is 25 (0x19).
- Opaque data in the object ID is the StorageGRID UUID prepended by the object-type byte. The object-type byte for data objects is `0x00`.
- General form of a CDMI object ID is as follows: `00006FFD0019`*`CRC`*`00`*`UUID`*.

  Where *`CRC`* is the cyclic redundancy check (CRC) number in hexadecimal, and where *`UUID`* is the StorageGRID UUID number without dashes.

The following Ruby function converts a StorageGRID UUID to a CDMI object ID:

```
# Convert UUID to ObjectID
def to_oid(uuid)
  uuid = uuid.delete("-")
  bytes = [0, 0, 111, 253, 0, 25, 0, 0, 0] + hexToBytes(uuid)
  crc = crc16(bytes)
  bytes[6],bytes[7] = crc >> 8, crc & 0xFF
  bytes.collect{|x| x.to_s(16).rjust(2,'0')}.join
end
```

The conversion function uses a library that converts the UUID hexadecimal string into an array of decimal representations for each byte. The following example of the library is in Ruby:

```
# Convert the UUID hex string into array of decimal representations of
# each byte
def hexToBytes(hex)
  result = []
  len = hex.length
  for i in (0..len-2).step(2)
    result.push((hex[i].chr + hex[i+1].chr).to_i(16))
  end
  return result
end
```

The conversion function also uses a library that calculates CRC. The following example of the library is in Ruby:

```
# Calculate the CRC
def crc16(bytes)
  result = 0
  bytes = bytes + [ 0, 0 ]
  bytes.each do |byte|
    byte.to_s(2).rjust(8, '0').reverse.each_char do |bit|
      overflow = (result & 0x8000) > 0
      result = ((result << 1) | (bit.to_i(2))) & 0xFFFF
      if overflow
        result = result ^ 0x8005
      end
    end
  end
  result.to_s(2).rjust(16, '0').reverse.to_i(2)
end
```

The following Ruby code converts a CDMI object ID to a StorageGRID UUID:

```
# Convert ObjectID to UUID
def to_uuid(oid)
  oid[18..25] + '-' + oid[26..29] + '-' + oid[30..33] + '-' + oid[34..37]
+ '-' + oid[38..-1]
end
```

# How CDMI clients use HTTP with the StorageGRID system

CDMI clients use the HTTP protocol to communicate with the StorageGRID system over a network connection that uses Transport Layer Security (TLS).

## Supported HTTP version

The StorageGRID system supports HTTP version 1.1.

For more information about HTTP, see HTTP/1.1 (RFC 2616).

### Related information

*http://www.ietf.org/rfc/rfc2616.txt*

## Default HTTP ports for the CLB and LDR services

The StorageGRID system includes default ports for the CLB service and the LDR service for clients to use to ingest, query, and retrieve objects.

| Grid service | Purpose | Default port number |
|---|---|---|
| CLB | Query and retrieve | 8080 |
| | Ingest | 8081 |
| LDR | Query and retrieve | 18080 |
| | Ingest | 18081 |

The grid might be configured with default ports or customized ports. You can view the ports as configured in the grid in the NMS MI under **Grid Management > Grid Configuration > Storage > Main**

It is recommended that you use the default HTTP ports for their intended purposes to maintain grid efficiency. For example, as a grid matures some LDR services and CMS services fill up and become read-only. When the grid directs queries to an ingest port, the CLB service directs queries to resources that support both read and write operations, not resources that support read operations. An ingest request sent to the query/retrieve port might fail when the grid directs the query to an LDR that is read-only.

**Related tasks**

*Finding port numbers for the LDR service and the CLB service* on page 25

# How the StorageGRID system implements security

The StorageGRID system only accepts HTTP commands submitted over a network connection that uses Transport Layer Security (TLS) to provide application authentication and, optionally, transport encryption.

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

## Supported hashing and encryption algorithms for TLS libraries

The Transport Layer Security (TLS) libraries used by the StorageGRID system support a limited set of hashing and encryption algorithms that clients can use when establishing a TLS session with the grid.

The StorageGRID system supports the following cipher suites:

- AES128-SHA
- AES256-SHA
- NULL-SHA
- NULL-MD5

Based on system measurements and general security domain knowledge, AES128-SHA and AES256-SHA provide reasonable security without requiring inordinate amounts of computational resources. The choice between AES128-SHA and AES256-SHA depends on the requirements for the client application that balance performance and encryption security.

**Note:** It is recommended that you use one of the NULL ciphers if encryption is not required, and you want to eliminate the overhead associated with encryption. The client must explicitly request the NULL cipher.

## How CDMI clients use certificates for security

When a CDMI client establishes a TLS session to the grid, the grid sends a server certificate to the CDMI client for verification to ensure that the HTTP connection is secure.

This certificate can be verified using the grid HTTP certificate that was generated during the grid installation. The client application loads the grid HTTP certificate and uses it to verify that the client application is communicating with the expected grid. This process protects against man-in-the-middle and impersonation attacks.

**Note:** The Common Name (CN) field in the SSL server certificate that the grid returns to the client has a node ID as its value rather than the host name or IP address of the server. The client should

not perform host name verification on the CN in the server certificate because the verification will fail.

It is recommended that CDMI clients send client certificates to the grid as part of the session establishment process. A certificate is required if the CDMI client is assigned to a security partition or if the client's assigned HTTP profile requires certificate authentication. This certificate must be loaded into the StorageGRID system as part of the StorageGRID configuration process. For more information, see the *StorageGRID API Guide*.

**Related tasks**

*Copying the CA certificate from the StorageGRID system* on page 23

# Time synchronization between CDMI clients and the grid

You should synchronize time between CDMI clients and the StorageGRID system to maintain reliability and security.

The StorageGRID system uses NTP to synchronize the clock on the Gateway Node with the selected NTP server.

The grid associates a timestamp with each HTTP transaction that it performs. You can convert the UTC timestamp to local time, and compare the time reported by the grid to the time reported by the CDMI client to determine whether the CDMI client uses the correct time.

CDMI clients can use the StorageGRID system time to provide traceable timestamps for events and transactions, and the timestamps can enable temporal correlation in security audit logs.

# Best practices for HTTP sessions

How you configure idle, active, and concurrent HTTP sessions can impact grid performance. A number of best practices help you configure HTTP sessions in the most efficient manner.

## HTTP session duration

Best practices for HTTP session duration can help optimize grid performance.

### Best practices for idle HTTP sessions

You should keep HTTP sessions open even when client applications are idle to allow client applications to perform subsequent transactions over the open session.

Open and idle HTTP sessions provide the following benefits:

- Reduced latency from the time that the grid determines it has to perform an HTTP transaction to the time that the grid 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 grid

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

Based on system measurements and integration experience, you should keep an HTTP session open for a maximum of 10 minutes. The LDR service might automatically close an HTTP session that is kept open and idle for longer than 10 minutes.

### Best practices for active HTTP sessions

You should limit the duration of an HTTP session, even if the HTTP session continuously performs transactions.

Bounded HTTP sessions provide the following benefits:

- Enables optimal load balancing across the grid
- Allows maintenance procedures to start

  Some maintenance procedures only start after all the in-progress HTTP sessions are complete.
- Allows client applications to direct HTTP transactions to LDR services and ARC services that have available space

To optimize load balancing across the grid, you should prevent long-lived TCP/IP connections. You should configure client applications to track the age of each HTTP session and close the HTTP session after a set time so that the HTTP session can be re-established and re-balanced.

The grid balances its load when a client application establishes an HTTP session. Over time, an HTTP session used by the grid for a compute resource might no longer be optimal as load balance requirements change. The grid performs its best load balancing when client applications establish a separate HTTP session for each transaction, but this negates the much more valuable gains associated with persistent sessions.

Determining the maximum duration that a session should be held open is a trade-off between the benefits of session persistence and the ideal allocation of the session to internal grid resources. Based on system measurements and integration experience, it is recommended to keep a session open for a maximum of 10 minutes.

## Best practices for concurrent HTTP sessions

You must keep multiple TCP/IP connections to the grid open to allow idle sessions to perform transactions as required. The number of clients also affects how you handle multiple TCP/IP connections.

Concurrent HTTP sessions provide the following benefits:

- Reduced latency
  Transactions can start immediately instead of waiting for other transactions to be completed.
- Increased throughput
  The grid can perform parallel transactions and increase aggregate transaction throughput.

It is recommended that clients establish multiple HTTP sessions, either on a client-by-client basis or on a session-pool basis. When a client has to perform a transaction, it can select and immediately use any established session that is not currently processing a transaction.

Each grid topology has different peak throughput for concurrent transactions and sessions 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 supported by the grid are also factors.

Based on system measurements and integration experience, the recommended maximum number of concurrent sessions that each instance of a client should keep established at any given time is 50 per Gateway Node.

Each Gateway Node can support a maximum of 450 concurrent sessions. In small grid configurations, performance degrades when more than 50 sessions perform HTTP transactions at the same time.

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

- The grid topology affects the maximum number of concurrent transactions and sessions that the grid can support.
- Client applications that interact with the grid 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 grid, you might have to reduce the degree of concurrency to avoid exceeding the limits of the grid.

Client applications can use concurrent HTTP sessions across multiple Gateway Nodes to balance loads. The grid benefits from concurrent HTTP sessions across multiple Gateway Nodes during normal operations and fault conditions. During normal operations, each Gateway Node handles a reduced subset of the sessions and data transfers across multiple independent network interfaces. During fault conditions, only a subset of the sessions is lost, reducing the disruption from the fault and allowing rapid recovery using already established sessions through the Gateway Nodes that were not affected.

Balancing loads across multiple Gateway Nodes is required to push the aggregate transaction throughput beyond 1 Gb/s.

## Pools of HTTP sessions for read and write

You can use separate pools of HTTP sessions for read and write operations and control how much of each pool to use for read and write operations.

Clients can create loads that are retrieve-dominant or store-dominant. With pools of HTTP sessions for read and write transactions, you can adjust how much of each pool to dedicate for read or write transactions. Pools of HTTP sessions enable you to better control transactions and balance loads.

It is recommended to establish sessions in the read pool to the query/retrieve port of the grid, and establish sessions for the write pool to the ingest port of the grid.

## How CDMI clients affect the HTTP transaction load of grids

Understanding the HTTP transaction profiles of CDMI clients can help you calculate the estimated transaction load on the grid and ensure that the grid can manage the transaction load.

Different CDMI clients have different HTTP transaction profiles. For example, a CDMI client that primarily stores content places an HTTP transaction load on the grid that consists mostly of `POST` transactions. A CDMI client that primarily retrieves content consists mostly of `GET` transactions.

When you design CDMI client integration with the grid, it is recommended that you plan and diagram the interaction sequences between the CDMI client and the grid to determine the transactions performed for each application-specific set of functionality. This lets you map the grid transaction coverage to application-specific operations.

Once you determine the HTTP transactions associated with each type of application-specific functionality, you can calculate the transaction load on the grid based on the use profiles of the

application. For example, if a CDMI client stores 100 new objects each hour, the `POST` rate is 0.027 transactions per second.

By providing a translation between application-specific tasks and the corresponding load on the grid, users of the application can size the grid based on their use of the application.

Many clients are designed to deploy multiple instances of the client to handle multiple sites, workgroups, or devices. When you deploy multiple instances of a client, the load numbers for the transactions should reflect the load placed on the grid by a single instance of the client. You can calculate the load placed on the grid by multiple instances of clients by adding the numbers for the application-specific loads, and then converting into the corresponding grid load.

# Copyright information

Copyright © 1994–2012 NetApp, Inc. All rights reserved. Printed in the U.S.

No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

# Trademark information

NetApp, the NetApp logo, Network Appliance, the Network Appliance logo, Akorri, ApplianceWatch, ASUP, AutoSupport, BalancePoint, BalancePoint Predictor, Bycast, Campaign Express, ComplianceClock, Cryptainer, CryptoShred, Data ONTAP, DataFabric, DataFort, Decru, Decru DataFort, DenseStak, Engenio, Engenio logo, E-Stack, FAServer, FastStak, FilerView, FlexCache, FlexClone, FlexPod, FlexScale, FlexShare, FlexSuite, FlexVol, FPolicy, GetSuccessful, gFiler, Go further, faster, Imagine Virtually Anything, Lifetime Key Management, LockVault, Manage ONTAP, MetroCluster, MultiStore, NearStore, NetCache, NOW (NetApp on the Web), Onaro, OnCommand, ONTAPI, OpenKey, PerformanceStak, RAID-DP, ReplicatorX, SANscreen, SANshare, SANtricity, SecureAdmin, SecureShare, Select, Service Builder, Shadow Tape, Simplicity, Simulate ONTAP, SnapCopy, SnapDirector, SnapDrive, SnapFilter, SnapLock, SnapManager, SnapMigrator, SnapMirror, SnapMover, SnapProtect, SnapRestore, Snapshot, SnapSuite, SnapValidator, SnapVault, StorageGRID, StoreVault, the StoreVault logo, SyncMirror, Tech OnTap, The evolution of storage, Topio, vFiler, VFM, Virtual File Manager, VPolicy, WAFL, Web Filer, and XBB are trademarks or registered trademarks of NetApp, Inc. in the United States, other countries, or both.

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. A complete and current list of other IBM trademarks is available on the web at *www.ibm.com/legal/copytrade.shtml*.

Apple is a registered trademark and QuickTime is a trademark of Apple, Inc. in the United States and/or other countries. Microsoft is a registered trademark and Windows Media is a trademark of Microsoft Corporation in the United States and/or other countries. RealAudio, RealNetworks, RealPlayer, RealSystem, RealText, and RealVideo are registered trademarks and RealMedia, RealProxy, and SureStream are trademarks of RealNetworks, Inc. in the United States and/or other countries.

All other brands or products are trademarks or registered trademarks of their respective holders and should be treated as such.

NetApp, Inc. is a licensee of the CompactFlash and CF Logo trademarks.

NetApp, Inc. NetCache is certified RealSystem compatible.

# How to send your comments

You can help us to improve the quality of our documentation by sending us your feedback.

Your feedback is important in helping us to provide the most accurate and high-quality information. If you have suggestions for improving this document, send us your comments by email to *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.

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

## A



## B



## C

## D



## E



## F



## G



## H

## I



## L



## M



## N



## O



## P