




# **EMC ATMOS System Management API**

**Version 1.3.0**

**Revision A**

**EMC CORPORATION**  
*Corporate Headquarters:*  
Hopkinton, MA 01748-9103  
1-508-435-1000  
[www.EMC.com](http://www.EMC.com)



© 2010 EMC Corporation. All rights reserved.

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

The information in this publication is provided “as is.” EMC Corporation makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

EMC is a registered trademark of EMC Corporation. All other trademarks used herein are the property of their respective owners.

# Contents

|  |           |
|--|-----------|
| <b>Preface</b> .....   | <b>6</b>  |
| About this Guide .....                                       | 6         |
| Typographic Conventions .....                                | 6         |
| Documentation .....  | 6         |
| <hr/>  |           |
| <b>PART I: ATMOS SYSTEM MANAGEMENT POX API</b>               |           |
| <b>1 System Management API</b> .....                         | <b>9</b>  |
| About the System Management API .....                        | 9         |
| Authentication .....   | 9         |
| Authenticate as a SysAdmin .....                             | 9         |
| Authenticate as a TenantAdmin .....                          | 10        |
| <b>2 Managing Tenants</b> .....                              | <b>12</b> |
| Working with Tenants and Tenant Admins .....                 | 12        |
| Obtaining the List of Tenants and Access Nodes .....         | 12        |
| Obtaining Information about One Tenant .....                 | 13        |
| API Reference .....  | 18        |
| Assign a Tenant Admin .....                                  | 18        |
| Create a Tenant .....  | 19        |
| Get a Tenant .....   | 20        |
| List Tenants .....   | 22        |
| Remove a Tenant Admin .....                                  | 24        |
| <b>3 Managing Subtenants, UIDs, and Shared Secrets</b> ..... | <b>26</b> |
| Working with Subtenants, UIDs, and Shared Secrets .....      | 26        |
| Obtaining the List of Subtenants .....                       | 26        |
| Obtaining Information about One Subtenant .....              | 27        |
| Subtenant API Reference .....                                | 29        |
| Assign a Subtenant Admin .....                               | 29        |
| Create a Subtenant .....                                     | 30        |
| Get a Subtenant .....  | 31        |
| List Subtenants .....  | 33        |
| Modify a Subtenant .....                                     | 34        |
| Remove a Subtenant Admin .....                               | 35        |

|   |           |
|---|-----------|
| UID API Reference .....                             | 36        |
| Create UID .....                                    | 37        |
| Delete UID .....                                    | 38        |
| Disable UID .....                                   | 39        |
| Enable UID .....                                    | 40        |
| Get UID .....                                       | 41        |
| List UIDs .....                                     | 42        |
| Shared Secret API Reference .....                   | 44        |
| Reset the Shared Secret .....                       | 44        |
| <b>4 Working with CIFS Shares .....</b>             | <b>46</b> |
| Working with CIFS Shares .....                      | 46        |
| Create a CIFS Node .....                            | 46        |
| Create a CIFS share .....                           | 46        |
| List all CIFS Shares for a Tenant/Subtenant .....   | 47        |
| API Reference .....                                 | 48        |
| Add CIFS Node .....                                 | 48        |
| Change CIFS Share .....                             | 50        |
| Create CIFS Share .....                             | 52        |
| Delete CIFS Shares .....                            | 54        |
| Global Configuration .....                          | 55        |
| List CIFS Shares .....                              | 57        |
| Translate Node Name to UUID .....                   | 58        |
| <b>5 Working with NFS Shares .....</b>              | <b>60</b> |
| Working with NFS Shares .....                       | 60        |
| Adding an NFS Node .....                            | 60        |
| Creating an NFS Share .....                         | 60        |
| Listing the NFS Shares for a Tenant/Subtenant ..... | 61        |
| API Reference .....                                 | 61        |
| Create NFS Node .....                               | 62        |
| Create NFS Shares .....                             | 64        |
| Change NFS Shares .....                             | 65        |
| Delete NFS Shares .....                             | 67        |
| List NFS Shares .....                               | 68        |
| Translate Node Name to UUID .....                   | 70        |
| <b>6 Working with the Atmos Policy API .....</b>    | <b>71</b> |
| About Policy .....                                  | 71        |
| Working with Policies .....                         | 72        |
| Creating a Policy Specification .....               | 72        |

|  |     |
|--|-----|
| Policy Selectors API Reference .....     | 79  |
| Assign a Policy Selector .....           | 80  |
| Create a Policy Selector .....           | 81  |
| Delete a Policy Selector .....           | 84  |
| Get a Policy Selector .....              | 85  |
| List Policy Selectors .....              | 86  |
| Modify a Policy Selector .....           | 88  |
| Remove a Policy Selector .....           | 91  |
| Policy Specification API Reference ..... | 92  |
| Create a Policy Specification .....      | 92  |
| Delete a Policy Specification .....      | 99  |
| Get a Policy Specification .....         | 100 |
| List Policy Specifications .....         | 102 |
| Modify a Policy Specification .....      | 104 |

---

## **PART II: ATMOS SYSTEM MONITORING REST API**

|          |   |            |
|----------|---|------------|
| <b>7</b> | <b>System Monitoring API .....</b>                | <b>113</b> |
|          | About the System Monitoring API .....             | 113        |
|          | Authentication, Data Integrity, and Privacy ..... | 113        |
|          | API Reference .....                               | 114        |
|          | RMG List .....                                    | 114        |
|          | RMG Info .....                                    | 115        |
|          | Nodes List .....                                  | 117        |
|          | Nodes info .....                                  | 118        |
|          | Web Service Statistics .....                      | 119        |
|          | Atmos Storage Consumption .....                   | 121        |
|          | Error Responses .....                             | 122        |

# Preface

---

## About this Guide

This document describes the EMC<sup>®</sup> Atmos<sup>™</sup> system management API. The system management API allows you to programmatically access and control a subset of system management functions that are available in the system management GUI. The system management API is a combination of plain old XML (POX) and RESTful operations. Atmos plans to replace the POX-based API with a RESTful API over time.

To ensure an XML response, the `Accept` HTTP header must be set to `application/xml`.

This documentation includes sample request and response data. You must modify the parameter values for your own application.

---

## Typographic Conventions

The following conventions are used in this guide:

---

| Conventions               | Meaning                                |
|---------------------------|--|
| <a href="#">Blue text</a> | Hyperlinked cross reference or URL.    |
| <code>FixedWidth</code>   | Commands, filenames, and code examples |
| <b>FixedWidthBold</b>     | Emphasis within code examples          |
| <i>FixedWidthItalics</i>  | Variable names in text                 |

---

---

## Documentation

The following documents comprise the Atmos documentation set. You can find them at the EMC Powerlink Web site ([powerlink.emc.com](http://powerlink.emc.com)). After logging in, select Support > Technical Documentation and Advisories, then the links to software or hardware documentation for Atmos.

---

| Document                             | Description  | Intended Audience                         |
|--------------------------------------|--|---|
| <i>EMC Atmos Release Notes</i>       | Presents summary information about changes in a given release. | Anyone preparing for a new Atmos release. |
| <i>EMC Atmos Conceptual Overview</i> | Presents a high-level overview of Atmos.                       | All Atmos users.                          |

---

| <b>Document</b>                                      | <b>Description</b>   | <b>Intended Audience</b>   |
|--|--|--|
| <i>EMC Atmos Installation and Upgrade Guide</i>      | Explains how to install, configure, and upgrade Atmos on VMware and physical hardware as well as several optional RPMs.  | Users who install and configure Atmos.   |
| <i>EMC Atmos System Administrator's Guide</i>        | Describes how to use the system-management GUI to access and manage Atmos server software.   | Atmos system managers. Also background for anyone who installs Atmos software. |
| <i>EMC Atmos Programmer's Guide</i>                  | Explains how to use the Atmos Web-service APIs to create objects and manage their related metadata.  | Developers who build or deploy applications using the APIs.                    |
| <i>EMC Atmos System Management API Guide</i>         | Describes how to use the Atmos System Management API to programmatically perform a subset of system management tasks.  | Developers who build or deploy applications using the APIs.                    |
| <i>EMC Atmos Security Configuration Guide</i>        | Provides an overview of settings available to ensure the secure operation of Atmos.  | Atmos system managers.   |
| <i>EMC Atmos Non-EMC Software License Agreements</i> | Presents legal notices for non-EMC software that is available through Atmos.   | —  |
| <i>EMC Atmos Hardware Guide</i>                      | Describes the hardware which is the foundation of the Atmos platform.<br><br>This document has a different numbering scheme than the other, software manuals, because it has a different release schedule. | Anyone who installs and maintains Atmos hardware.                              |



**PART I:**

**ATMOS SYSTEM MANAGEMENT  
POX API**

# 1 System Management API

This chapter describes the EMC® Atmos™ system management API. It includes the following topics:

- ▶ [About the System Management API](#)
- ▶ [Authentication](#)

---

## About the System Management API

The Atmos system management API returns XML. You can use this API to:

- ▶ Manage tenants and subtenants.
- ▶ Assign Atmos nodes as NFS or CIFS, then perform share management functions.
- ▶ Define policies programmatically.

---

## Authentication

The system management operations require an authenticated session. This section describes the methods to:

- ▶ [Authenticate as a SysAdmin](#)
- ▶ [Authenticate as a TenantAdmin](#)

## Authenticate as a SysAdmin

|                           |  |
|---------------------------|--|
| <b>Description</b>        | Authenticates a user in the Atmos SysAdmin role. Returns a session ID ( <code>_gui_session_id</code> ) in the HTTP response header <b>Set-Cookie</b> . All subsequent requests using this API must include this session ID in the HTTP request header <b>Cookie</b> . You pass the SysAdmin's authentication credentials in the HTTP request body using the <code>username</code> and <code>password</code> parameters. The username/password credentials must exist and be for a user in the SysAdmin role. Returns a Boolean representing success ( <code>true</code> ) or failure (an error message) of the authentication request. There is no logout request. An inactive session expires after 30 minutes. |
| <b>HTTP Method</b>        | POST   |
| <b>URI</b>                | <code>/mgmt_login/verify</code>  |
| <b>Request Parameters</b> | HTTP Body:   |

- ▶ **auth\_type**: Specify the value `local`.
- ▶ **auth\_addr**: Specify the IP address/host name of the remote location where the specified SysAdmin user is authenticating from; this is only applicable if you specify `remote` **auth\_type**.
- ▶ **username**: Specify the username of a SysAdmin user.
- ▶ **password**: Specify the password for the SysAdmin user.

**Request Header**

```
POST /mgmt_login/verify HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Host: 10.5.116.110
Content-Length: 64
```

**Request Body**

```
auth_type=local&auth_addr=&username=SysAdmin&password=%231Passwd
```

**Response Header**

```
HTTP/1.1 200 OK
Date: Fri, 11 Sep 2009 13:22:52 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 36
Set-Cookie: _gui_session_id=113fa69bb6173fa6016a75bbdc3fc06a; path=/
Connection: close
```

**Response Body**

```
<authenticated>true</authenticated>
```

## Authenticate as a TenantAdmin

**Description**

Authenticates an Atmos tenant administrator. Returns a session ID (`_gui_session_id`) in the HTTP response header **Set-Cookie**. All subsequent requests using this API must include this session ID in the HTTP request header **Cookie**. You pass the tenant administrator's authentication credentials in the HTTP request body using the `tenant_name`, `username`, and `password` parameters. The username/password credentials must exist and be for a user in the tenant administrator role. Returns a Boolean representing success (`true`) or failure (an error message) of the authentication request. There is no logout request. An inactive session expires after 30 minutes.

**HTTP Method**

POST

**URI**

`/user/verify`

**Request parameters**

HTTP body:

- ▶ **tenant\_name**: Specify the name of an existing Atmos tenant.
- ▶ **username**: Specify the Atmos tenant administrator username.
- ▶ **password**: Specify the password for the Atmos tenant administrator.

**Request Header**

```
POST /user/verify HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Host: host:port
Content-Length: 68
```

**Request Body**

```
tenant_name=maas_tenant&username=maas_tenant_admin&password=password
```

**Response Header**

```
HTTP/1.1 200 OK
Date: Mon, 06 Jul 2009 17:31:07 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 36
Set-Cookie: _gui_session_id=1f856413a34c20aba0ddb5dc0206fa35; path=/
Connection: close
```

**Response Body**

A successful login returns:

```
<authenticated>true</authenticated>
```

An unsuccessful login returns:

```
<authenticated>Authentication failed</authenticated>
```

# 2 Managing Tenants

This section describes the API for managing tenants and tenant administrators. It includes the following topics:

- ▶ [Working with Tenants and Tenant Admins](#)
- ▶ [API Reference](#)

All of the API calls require an authenticated session. To create an authenticated session, see [Chapter , “Authentication” on page 9](#).

---

## Working with Tenants and Tenant Admins

This section describes the set of commands you use for:

- ▶ [Obtaining the List of Tenants and Access Nodes](#)
- ▶ [Obtaining Information about One Tenant](#)

### Obtaining the List of Tenants and Access Nodes

To obtain the details about the tenants for the authenticated use in the SysAdmin role:

- 1 Authenticate as an Atmos SysAdmin. For more information, see [“Authenticate as a SysAdmin” on page 9](#).
- 1 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 2 Use the HTTP GET command with this URI:

```
/maui_admin/list_tenant
```

The following body shows an example of what is returned:

```
<tenant_list>
  <tenant>
    <name>t1</name>
    <id>6387df4a05e14bdea1a20e410d2ebcda</id>
    <status>Initialized</status>
    <authentication_source>Local</authentication_source>
```

```

<tenant_admin_list>
  <tenant_admin>t1admin</tenant_admin>
  <tenant_admin>testadmin</tenant_admin>
</tenant_admin_list>
<access_node_list>
  <access_node>IS01-002</access_node>
  <access_node>IS01-003</access_node>
  <access_node>IS01-001</access_node>
  <access_node>IS01-004</access_node>
</access_node_list>
</tenant>
<tenant>
  <name>t2</name>
  <id>0e27ce6cc0884d7ca3be90a5f69e3291</id>
  <status>Initialized</status>
  <authentication_source>Local</authentication_source>
  <tenant_admin_list>
    <tenant_admin>t2admin</tenant_admin>
    <tenant_admin>Test</tenant_admin>
  </tenant_admin_list>
  <access_node_list>
    <access_node>No access node</access_node>
  </access_node_list>
</tenant>
</tenant_list>

```

## Obtaining Information about One Tenant

To obtain the details about the tenants for the authenticated use in the SysAdmin role:

- 1 Authenticate as an Atmos SysAdmin. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).

- 2 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 3 Use the HTTP GET command with this URI:

```
/tenant_admin/get_tenant_info
```

The following body shows an example of what is returned:

```
<tenant>
  <id>6387df4a05e14bdea1a20e410d2ebcda</id>
  <name>t1</name>
  <status>Initialized</status>
  <authentication_source>Local</authentication_source>
  <tenant_admin_list>
    <tenant_admin>
      <name>t1admin</name>
      <authentication_source>Local</authentication_source>
    </tenant_admin>
    <tenant_admin>
      <name>test</name>
      <authentication_source>Local</authentication_source>
    </tenant_admin>
  </tenant_admin_list>
  <policy_distribution_status>Completed</policy_distribution_status>
  <access_node_list>
    <access_node>
      <name>IS01-002</name>
      <id>564D1DDC-8B85-5B06-BA35-F0A26889C6B6</id>
      <public_ip>10.5.116.245</public_ip>
      <webservice>disable</webservice>
      <filesystem>cifs</filesystem>
      <multi_subtenant_access>disable</multi_subtenant_access>
    </access_node>
```

```

<access_node>
  <name>IS01-003</name>
  <id>564D188C-F514-7901-39E3-63B2CDCA527A</id>
  <public_ip>10.5.116.246</public_ip>
  <webservice>enable</webservice>
  <filesystem>nfs</filesystem>
  <multi_subtenant_access>disable</multi_subtenant_access>
</access_node>
<access_node>
  <name>IS01-001</name>
  <id>564D85B4-8FA2-34BF-24A4-9BFB9C17A02B</id>
  <public_ip>10.5.116.244</public_ip>
  <webservice>disable</webservice>
  <filesystem>nfs</filesystem>
  <multi_subtenant_access>disable</multi_subtenant_access>
</access_node>
<access_node>
  <name>IS01-004</name>
  <id>564DF58B-B8B3-EC98-7F8B-E23B098FC976</id>
  <public_ip>10.5.116.247</public_ip>
  <webservice>enable</webservice>
  <filesystem>nfs</filesystem>
  <multi_subtenant_access>disable</multi_subtenant_access>
</access_node>
</access_node_list>
<capacity>0</capacity>
<sub_tenant_list>
  <sub_tenant>
    <name>t1</name>
    <id>6387df4a05e14bdea1a20e410d2ebcda</id>

```

```

<status>Initialized</status>

<authentication_source>Local</authentication_source>

<sub_tenant_admin_list>
  <sub_tenant_admin>t1admin</sub_tenant_admin>
</sub_tenant_admin_list>
</sub_tenant>
<sub_tenant>
  <name>t1admin</name>
  <id>ad33e4c978c74dfbb31d5c6363b13ec4</id>
  <status>Initialized</status>
  <authentication_source>Local</authentication_source>
  <sub_tenant_admin_list>
    <sub_tenant_admin>No SubTenant Admin</sub_tenant_admin>
  </sub_tenant_admin_list>
</sub_tenant>
</sub_tenant_list>
<policy_list>
  <policy>
    <name>default</name>
    <expression></expression>
    <policy_id>0</policy_id>
    <replica_list>
      <replica>
        <type>sync</type>
        <storage_mechanism></storage_mechanism>
        <location></location>
      </replica>
      <replica>
        <type>sync</type>
        <storage_mechanism></storage_mechanism>

```

```

        <location></location>
    </replica>
</replica_list>
    <retention></retention>
    <deletion></deletion>
</policy>
</policy_list>
<policy_selector_list>
    <policy_selector>
        <name>policyselector1</name>
        <expression>atime >= 2010-01-08 00:00:00</expression>
        <on_event>ON_CREATE</on_event>
        <spec>default</spec>
    </policy_selector>
    <policy_selector>
        <name>policyselector3</name>
        <expression>itime equals 2010-01-25</expression>
        <on_event>ON_CREATE</on_event>
        <spec>Policy4</spec>
    </policy_selector>
    <policy_selector>
        <name>policyselector4</name>
        <expression>itime equals 2010-01-25</expression>
        <on_event>ON_CREATE</on_event>
        <spec>Policy3</spec>
    </policy_selector>
</policy_selector_list>
<handler_list>
</handler_list>
<export_list>

```

```
</export_list>
</tenant>
```

---

## API Reference

- ▶ [Assign a Tenant Admin](#)
- ▶ [Create a Tenant](#)
- ▶ [Get a Tenant](#)
- ▶ [List Tenants](#)
- ▶ [Remove a Tenant Admin](#)

---

### Assign a Tenant Admin

|                           |  |
|---------------------------|--|
| <b>Description</b>        | Adds a user to the TenantAdmin role for the specified tenant. Returns true if successful. The HTTP request header must include the <code>Cookie</code> with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication. The HTTP request header must include the <code>Accept</code> with the value set to <code>application/xml</code> .   |
| <b>Required Role</b>      | SysAdmin   |
| <b>HTTP Method</b>        | GET  |
| <b>URI</b>                | <code>/maui_admin/submit_tenant_admin_info</code>  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"><li>▶ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li><li>▶ <b>Accept:</b> Set to <code>application/xml</code>.</li></ul> <p>Querystring parameters:</p> <ul style="list-style-type: none"><li>▶ <b>operation:</b> Specify the operation type to run; for this request, the value is always <code>add</code>.</li><li>▶ <b>flag_new:</b> Specify if user is new (value is <code>on</code>) or not (value is <code>off</code>).</li><li>▶ <b>tenant_name:</b> Specify the name of the tenant.</li><li>▶ <b>username:</b> Specify the name of the user to be assigned to the TenantAdmin role.</li><li>▶ <b>password:</b> Specify the password for the new user. This is only valid when <b>flag_new</b> is set to <code>on</code>.</li><li>▶ <b>password_confirm:</b> Confirm the password for the new user. Only valid when <b>flag_new</b> is set to <code>on</code>.</li></ul> |

**Request Header** GET  
/maui\_admin/submit\_tenant\_admin\_info?operation=add&flag\_new=off&tenant\_name=TestTenant&username=TestAdmin&password=&password\_confirm= HTTP/1.1  
Accept: application/xml  
Cookie: \_gui\_session\_id=5cff6a5144b259dee629d1d204d3aceb  
Host: 10.5.116.110

**Response Header** HTTP/1.1 200 OK  
Date: Fri, 11 Sep 2009 13:22:38 GMT  
Server: Mongrel 1.1.3  
Status: 200 OK  
=Cache-Control: no-cache  
Content-Type: application/xml; charset=utf-8  
Content-Length: 24  
Set-Cookie: \_gui\_session\_id=5cff6a5144b259dee629d1d204d3aceb; path=/  
Connection: close

**Response Body** A successful response looks like this:  
  
<cleared>true</cleared>

---

## Create a Tenant

**Description** Creates an Atmos tenant. Returns a tenant ID. The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication. The request body must specify the new tenant's name, creation type, and authorization type. When you create a tenant, Atmos creates a default subtenant automatically. The default subtenant has the same name, ID, and authentication source.

**Required Role** SysAdmin

**HTTP Method** POST

**URI** /maui\_admin/add\_tenant

**Request Parameters** HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.
- ▷ **Accept:** Set to `application/xml`.

HTTP body:

- ▷ **auth\_type:** Specify `local`. Specifies how the tenant is authenticated.
- ▷ **tenant\_name:** Specify a string representing the new tenant. It must be unique, and 30 characters or less.
- ▷ **creation\_type:** Specify `sync`.

**Request Header**  
POST /maui\_admin/add\_tenant HTTP/1.1  
Accept: application/xml  
Content-Type: application/x-www-form-urlencoded  
Cookie: \_gui\_session\_id=fa6818a982d7e35a8aee6c9f8b639e82  
Host: 10.5.116.110  
Content-Length: 61

**Request Body**  
auth\_type=local&tenant\_name=testtenant&creation\_type=sync

**Response Header**  
HTTP/1.1 200 OK  
Date: Thu, 10 Sep 2009 13:29:11 GMT  
Server: Mongrel 1.1.3  
Status: 200 OK  
Cache-Control: no-cache  
Content-Type: application/xml; charset=utf-8  
Content-Length: 56  
Set-Cookie: \_gui\_session\_id=fa6818a982d7e35a8aee6c9f8b639e82; path=/  
Connection: close

**Response Body**  
<tenant\_id>8bdb8ce67eaf4a4f946d24cce76c9974</tenant\_id>

---

## Get a Tenant

Returns an XML document containing information about the tenant associated with the TenantAdmin making the request. The information includes:

- ▶ Tenant name
- ▶ List of TenantAdmins
- ▶ List of subtenants and subtenant admins
- ▶ List of Web Services access nodes
- ▶ List of CIFS and NFS access nodes
- ▶ Policies

The HTTP request header must include the Cookie with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role** TenantAdmin

**HTTP Method** GET

**URI** /tenant\_admin/get\_tenant\_info

**Request Parameters** HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

▷ **Accept:** Set to application/xml.

### Request Header

GET /tenant\_admin/get\_tenant\_info HTTP/1.1  
Accept: application/xml  
Cookie: \_gui\_session\_id=2948ab711e7c190d37ebc201d2bc2169  
Host: 10.5.116.110

### Response Header

HTTP/1.1 200 OK  
Date: Fri, 11 Sep 2009 11:58:23 GMT  
Server: Mongrel 1.1.3  
Status: 200 OK  
Cache-Control: no-cache  
Content-Type: application/xml; charset=utf-8  
Content-Length: 1514  
Set-Cookie: \_gui\_session\_id=2948ab711e7c190d37ebc201d2bc2169; path=/  
Connection: close

### Response Body

```
<tenant>
  <id>478e1c71b39843dabfada070b21c851a</id>
  <name>TestTenant</name>
  <status>Initialized</status>
  <authentication_source>Local</authentication_source>
  <tenant_admin_list>
    <tenant_admin>
      <name>TestAdmin</name>
      <authentication_source>Local</authentication_source>
    </tenant_admin>
  </tenant_admin_list>
  <ws_access_node_list>
    <ws_access_node>
      <name>Testtest1-002</name>
      <id>564D3161-E062-BDD7-869F-296382B5F712</id>
      <public_ip>10.5.116.111</public_ip>
      <status>Service up</status>
    </ws_access_node>
    <ws_access_node>
      <name>Testtest1-001</name>
      <id>564DE0F3-A5BF-9DFD-8C5B-5EA182150521</id>
      <public_ip>10.5.116.110</public_ip>
      <status>Service up</status>
    </ws_access_node>
  </ws_access_node_list>
  <nfs_cifs_node_list>
    <nfs_cifs_node>
      <name>Testtest2-001</name>
      <id>564D4F54-A020-FAB6-51C5-95E59E8D971E</id>
      <type>nfs</type>
      <public_ip>10.5.116.112</public_ip>
      <status>Service up</status>
    </nfs_cifs_node>
    <nfs_cifs_node>
      <name>Testtest2-002</name>
      <id>564DC4E7-B0F3-D0E7-BD28-06FFDC4B4D71</id>
      <type>cifs</type>
      <public_ip>10.5.116.113</public_ip>
      <status>Service up</status>
    </nfs_cifs_node>
  </nfs_cifs_node_list>
  <capacity>0</capacity>
  <sub_tenant_list>
    <sub_tenant>
```

```

    <name>TestTenant</name>
    <id>478e1c71b39843dabfada070b21c851a</id>
    <status>Initialized</status>
    <authentication_source>Local</authentication_source>
    <sub_tenant_admin_list>
      <sub_tenant_admin>TestSubAdmin</sub_tenant_admin>
    </sub_tenant_admin_list>
  </sub_tenant>
</sub_tenant_list>
<policy_list>
  <policy>
    <name>default</name>
    <expression></expression>
    <policy_id>0</policy_id>
    <replica_list>
      <replica>
        <type>sync</type>
        <storage_mechanism>OPTIMAL</storage_mechanism>
        <location>TestLoc1</location>
      </replica>
      <replica>
        <type>sync</type>
        <storage_mechanism></storage_mechanism>
        <location></location>
      </replica>
    </replica_list>
    <retention>4D</retention>
    <deletion>1M</deletion>
  </policy>
</policy_list>
<policy_selector_list>
  <policy_selector>
    <name>golmanselect</name>
    <expression>uid equals test</expression>
    <on_event>ON_CREATE</on_event>
    <spec>default</spec>
  </policy_selector>
</policy_selector_list>
<handler_list>
  <handler>
    <handle>10.5.116.112:8088</handle>
    <expression>name equals test</expression>
    <handle_id>Testhandler</handle_id>
    <on_event>ON_CREATE</on_event>
  </handler>
</handler_list>
<export_list>
  <export>
    <ip_address>10.5.116.113</ip_address>
  </export>
</export_list>
</tenant>

```

---

## List Tenants

**Description** Returns an XML document that contains general information about the tenants for SysAdmin making the request. The XML document contains the following information:

- ▶ Tenant names and UUIDs
- ▶ Tenant status
- ▶ TenantAdmin lists
- ▶ Web service access nodes assigned to each tenant.

Returns an empty XML document if there are no tenants. The HTTP request header must include the Cookie with the session ID (`_gui_session_id`) returned from a successful authentication. It must also include the `Accept` header with the value of `application/xml`.

**Required Role**

SysAdmin

**HTP Method**

GET

**URI**

`/maui_admin/list_tenant`

**Request Parameters**

HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.
- ▷ **Accept:** Set to `application/xml`.

**Request Header**

```
GET /maui_admin/list_tenant HTTP/1.1
Accept: application/xml
Cookie: _gui_session_id=57a4b0e5d01fb7c7a2e18a3711def36a
Host: 10.5.116.110
```

**Response Header**

```
HTTP/1.1 200 OK
Date: Fri, 11 Sep 2009 11:11:30 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 825
Set-Cookie: _gui_session_id=57a4b0e5d01fb7c7a2e18a3711def36a; path=/
Connection: close
```

## Response Body

```
<tenant_list>
  <tenant>
    <name>TestTenant2</name>
    <id>e60d9f18fc9a4f0d8babb7cf67e57e55</id>
    <status>Initialized</status>
    <authentication_source>Local</authentication_source>
    <tenant_admin_list>
      <tenant_admin>No Tenant Admin</tenant_admin>
    </tenant_admin_list>
    <ws_access_node_list>
      <ws_access_node>No access node</ws_access_node>
    </ws_access_node_list>
  </tenant>
  <tenant>
    <name>TestTenant3</name>
    <id>8bdb8ce67eaf4a4f946d24cce76c9974</id>
    <status>Initialized</status>
    <authentication_source>Local</authentication_source>
    <tenant_admin_list>
      <tenant_admin>No Tenant Admin</tenant_admin>
    </tenant_admin_list>
    <ws_access_node_list>
      <ws_access_node>No access node</ws_access_node>
    </ws_access_node_list>
  </tenant>
</tenant_list>
```

## Remove a Tenant Admin

|                           |  |
|---------------------------|--|
| <b>Description</b>        | Removes the specified user from the TenantAdmin role for the specified tenant. Returns true if successful. The HTTP request header must include the <code>Cookie</code> with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication. The HTTP request header must include the <code>Accept</code> with the value set to <code>application/xml</code> .  |
| <b>Required Role</b>      | SysAdmin   |
| <b>HTP Method</b>         | GET  |
| <b>URI</b>                | <code>/maui_admin/modify_tenant_admin</code>   |
| <b>Request Parameters</b> | HTTP header: <ul style="list-style-type: none"><li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li><li>▷ <b>Accept:</b> Set to <code>application/xml</code>.</li></ul> Querystring parameters: <ul style="list-style-type: none"><li>▶ <b>operation:</b> Specify the operation type to run; for this request, the value is always <code>delete</code>.</li><li>▶ <b>tenant_name:</b> Specify the tenant name for whom the user is in the TenantAdmin role.</li><li>▶ <b>username:</b> Specify the name of the user to be removed as a TenantAdmin of the specified tenant.</li></ul> |

**Request  
Header**

```
GET
/maui_admin/modify_tenant_admin?operation=delete&tenant_name=TestTenant&username=TestAdmin HTTP/1.1
Accept: application/xml
Cookie: _gui_session_id=113fa69bb6173fa6016a75bbdc3fc06a
Host: 10.5.116.110
```

**Response  
Header**

```
HTTP/1.1 200 OK
Date: Fri, 11 Sep 2009 13:22:52 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 24
Set-Cookie: _gui_session_id=113fa69bb6173fa6016a75bbdc3fc06a; path=/
Connection: close
```

**Response  
Body**

```
<cleared>>true</cleared>
```

# 3

## Managing Subtenants, UIDs, and Shared Secrets

This section describes the API for managing subtenants, UIDs, and shared secrets. It includes the following topics:

- ▶ [Working with Subtenants, UIDs, and Shared Secrets](#)
- ▶ [Subtenant API Reference](#)
- ▶ [UID API Reference](#)
- ▶ [Shared Secret API Reference](#)

All of the API calls require an authenticated session. To create an authenticated session, see [Chapter , “Authentication” on page 9](#).

---

### Working with Subtenants, UIDs, and Shared Secrets

A *subtenant* is a logical subset of a tenant that groups together policies, data access, and reporting capabilities. A *UID/shared secret* uniquely defines and registers an application within the Atmos system. The system restricts access to resources to registered applications.

This section describes the set of commands you use for:

- ▶ [Obtaining the List of Subtenants](#)
- ▶ [Obtaining Information about One Subtenant](#)

#### Obtaining the List of Subtenants

To obtain the list of subtenants for a TenantAdmin:

- 1 Authenticate as an Atmos TenantAdmin. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).
- 1 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 2 Use the HTTP GET command with this URI:

```
/tenant_admin/list_sub_tenant
```

The following body shows an example of what is returned:

```
<sub_tenant_list>
```

```

<sub_tenant>
  <name>t1</name>
  <id>6387df4a05e14bdea1a20e410d2ebcda</id>
  <authentication_source>Local</authentication_source>
  <status>Initialized</status>
  <sub_tenant_admin_list>
    <sub_tenant_admin>t1admin</sub_tenant_admin>
  </sub_tenant_admin_list>
</sub_tenant>
<sub_tenant>
  <name>t1admin</name>
  <id>ad33e4c978c74dfbb31d5c6363b13ec4</id>
  <authentication_source>Local</authentication_source>
  <status>Initialized</status>
  <sub_tenant_admin_list>
    <sub_tenant_admin>No Admin</sub_tenant_admin>
  </sub_tenant_admin_list>
</sub_tenant>
</sub_tenant_list>

```

## Obtaining Information about One Subtenant

- 1 Authenticate as an Atmos TenantAdmin. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).
- 3 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 4 Use the HTTP GET command with this URI:

```
/tenant_admin/sub_tenant_info?sub_tenant_name=t1
```

The following body shows an example of what is returned:

```

<sub_tenant>
  <id>6387df4a05e14bdea1a20e410d2ebcda</id>
  <name>t1</name>

```

```

<authentication_source>Local</authentication_source>
<status>Initialized</status>
<sub_tenant_admin_list>
  <sub_tenant_admin>
    <name>t1admin</name>
    <authentication_source>Local</authentication_source>
  </sub_tenant_admin>
</sub_tenant_admin_list>
<capacity>0</capacity>
<default_policy_spec>default</default_policy_spec>
<uid_secret_list>
  <uid_secret>
    <uid>doctest</uid>
    <shared_secret>mcejGBKsbfdGy17vszZcSvR+Kgw=</shared_secret>
    <status>enabled</status>
    <email>test1@emc.com</email>
  </uid_secret>
  <uid_secret>
    <uid>test1</uid>
    <shared_secret>7Sp4bZxPp49Gctf3q5xHk18fNJM=</shared_secret>
    <status>enabled</status>
    <email>test1@emc.com</email>
  </uid_secret>
</uid_secret_list>
<policy_selector_map_status>Completed</policy_selector_map_status>
<policy_selector_list>
  <policy_selector>
    <name>policyselector4</name>
    <expression>itime equals 2010-01-25</expression>
    <on_event>ON_CREATE</on_event>
  </policy_selector>
</policy_selector_list>

```

```
    <spec>Policy3</spec>
  </policy_selector>
  <policy_selector>
    <name>policyselector3</name>
    <expression>itime equals 2010-01-25</expression>
    <on_event>ON_CREATE</on_event>
  <spec>Policy4</spec>
</policy_selector>
</policy_selector_list>
<handler_list>
</handler_list>
<export_list>
</export_list>
</sub_tenant>
```

---

## Subtenant API Reference

- ▶ [Assign a Subtenant Admin](#)
- ▶ [Assign a Subtenant Admin](#)
- ▶ [Get a Subtenant](#)
- ▶ [List Subtenants](#)
- ▶ [Modify a Subtenant](#)
- ▶ [Remove a Subtenant Admin](#)

### Assign a Subtenant Admin

|                      |  |
|----------------------|--|
| <b>Description</b>   | Assigns the specified user to the SubtenantAdmin role for the specified subtenant. Returns true if successful or an error message if unsuccessful. The HTTP request header must include the <code>Cookie</code> with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication. It must also include the <code>Accept</code> header with the value of <code>application/xml</code> . |
| <b>Required Role</b> | TenantAdmin  |

|                           |   |
|---------------------------|---|
| <b>HTTP Method</b>        | POST  |
| <b>URI</b>                | /tenant_admin/submit_sub_tenant_admin_info  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> <li>▷ <b>Accept:</b> Set to <code>application/xml</code>.</li> </ul> <p>HTTP body:</p> <ul style="list-style-type: none"> <li>▶ <b>operation:</b> Specify the operation type; for this request, the value is always <b>add</b>.</li> <li>▶ <b>flag_new:</b> Specify if the user is new (value is <code>on</code>) or not (value is <code>off</code>).</li> <li>▶ <b>username:</b> Specify the name of the user to be assigned as the SubTenantAdmin for the subtenant.</li> <li>▶ <b>sub_tenant_name:</b> Specify the name of the subtenant.</li> </ul> |
| <b>Request Header</b>     | <pre>POST /tenant_admin/submit_sub_tenant_admin_info HTTP/1.1 Accept: application/xml Content-Type: application/x-www-form-urlencoded Cookie: _gui_session_id=c718f1f27ad152016c024787d4e07a80 Host: 10.5.116.110 Content-Length: 77</pre>  |
| <b>Request Body</b>       | <code>operation=add&amp;flag_new=off&amp;username=TestUser&amp;sub_tenant_name=TestTenant</code>  |
| <b>Response Header</b>    | <pre>HTTP/1.1 200 OK Date: Mon, 14 Sep 2009 19:07:41 GMT Server: Mongrel 1.1.3 Status: 200 OK Cache-Control: no-cache Content-Type: application/xml; charset=utf-8 Content-Length: 24 Set-Cookie: _gui_session_id=c718f1f27ad152016c024787d4e07a80; path=/ Connection: close</pre>  |
| <b>Response Body</b>      | <code>&lt;cleared&gt;&gt;true&lt;/cleared&gt;</code>  |

## Create a Subtenant

|                      |  |
|----------------------|--|
| <b>Description</b>   | Creates an Atmos subtenant within the tenant of the requesting tenant administrator. Returns a subtenant ID. The HTTP request header must include the <code>Cookie</code> with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication. The request body must specify the new subtenant's name, creation type, and authorization type. |
| <b>Required Role</b> | TenantAdmin  |
| <b>HTTP Method</b>   | POST   |

**URI** /tenant\_admin/add\_sub\_tenant

**Request Parameters** HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

HTTP body:

- ▷ **auth\_type:** Specify value as `local`.
- ▷ **sub\_tenant\_name:** Specify a string representing the new subtenant. This name must be unique within the tenant, and cannot exceed 30 characters.
- ▷ **creation\_type:** Specify the value as `sync`.

**Request Header** POST /tenant\_admin/add\_sub\_tenant HTTP/1.1  
Accept: application/xml  
Content-Type: application/x-www-form-urlencoded  
Cookie: \_gui\_session\_id=1f856413a34c20aba0ddb5dc0206fa35  
Host: host:port  
Content-Length: 57

**Request Body** auth\_type=local&sub\_tenant\_name=samplename&creation\_type=sync

**Response Header** HTTP/1.1 200 OK  
Date: Mon, 06 Jul 2009 17:31:07 GMT  
Server: Mongrel 1.1.3  
Status: 200 OK  
Cache-Control: no-cache  
Content-Type: application/xml; charset=utf-8  
Content-Length: 64  
Set-Cookie: \_gui\_session\_id=1f856413a34c20aba0ddb5dc0206fa35; path=/  
Connection: close

**Response Body** A successful request returns:

```
<sub_tenant_id>747015be2d454ccebcc81853ac7c465a</sub_tenant_id>
```

An unsuccessful request returns an error message, for example:

```
<sub_tenant_id>Error: Sub tenant samplename existing</sub_tenant_id>
```

## Get a Subtenant

**Description** Returns an XML document containing the following information about the specified subtenant:

- ▶ ID
- ▶ Name
- ▶ Authentication source
- ▶ Status

- ▶ Subtenant administrators
- ▶ Capacity
- ▶ Default policy specification
- ▶ Shared secrets for the subtenant's UIDs
- ▶ The export list

The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role**

TenantAdmin

**HTTP Method**

GET

**URI**

`/tenant_admin/sub_tenant_info`

**Request Parameters**

HTTP header:

- ▶ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

Querystring parameter:

- ▶ **sub\_tenant\_name:** Specify the name of the subtenant whose details you want.

**Request Header**

```
GET /tenant_admin/sub_tenant_info?sub_tenant_name=samplename HTTP/1.1
Accept: application/xml
Cookie: _gui_session_id=6807f31b8689a4bbf8a13110f7ba4f82
Host: host:port
```

**Request Body**

None

**Response Header**

```
HTTP/1.1 200 OK
Date: Mon, 06 Jul 2009 18:24:46 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 2086
Set-Cookie: _gui_session_id=6807f31b8689a4bbf8a13110f7ba4f82; path=/
Connection: close
```

**Response Body**

```
<sub_tenant>
  <id>747015be2d454ccebccc81853ac7c465a</id>
  <name>samplename</name>
  <authentication_source>Local</authentication_source>
  <status>Initialized</status>
  <sub_tenant_admin_list> </sub_tenant_admin_list>
  <capacity>0</capacity>
  <default_policy_spec>default</default_policy_spec>
  <uid_secret_list>
    <uid_secret>
      <uid>samplenameUid0</uid>
      <shared_secret>MSuDvtqmivSS4qRsvHyrAMuyY8Y=</shared_secret>
      <status>enabled</status>
      <email>a@b.com</email>
    </uid_secret>
    <uid_secret>
      <uid>samplenameUid1</uid>
      <shared_secret>FWUOt75mez2fGVU4AoyXHcMjQec=</shared_secret>
      <status>enabled</status>
      <email>a@b.com</email>
    </uid_secret>
  </uid_secret_list>
  <export_list> </export_list>
</sub_tenant>
```

## List Subtenants

**Description**

Returns an XML document that contains general information about all subtenants associated with the requesting tenant administrator. The XML document contains the following information about one or more subtenants:

- ▶ Subtenant name
- ▶ ID
- ▶ Authentication source
- ▶ Status
- ▶ Subtenant administrator name

Returns an empty XML document if there are no subtenants. The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role**

TenantAdmin

**HTTP Method**

GET

**URI**

/tenant\_admin/list\_sub\_tenant

**Request Parameters**

HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

**Request Header** GET /tenant\_admin/list\_sub\_tenant HTTP/1.1  
 Accept: application/xml  
 Cookie: \_gui\_session\_id=3cbab89e4a42a999f6f020a8e8e3a82a  
 Host: host:port

**Request Body** None

**Response Header** HTTP/1.1 200 OK  
 Date: Mon, 06 Jul 2009 17:37:29 GMT  
 Server: Mongrel 1.1.3  
 Status: 200 OK  
 Cache-Control: no-cache  
 Content-Type: application/xml; charset=utf-8  
 Content-Length: 2166  
 Set-Cookie: \_gui\_session\_id=3cbab89e4a42a999f6f020a8e8e3a82a; path=/  
 Connection: close

**Response Body** A successful request returns:

```

<sub_tenant_list>
  <sub_tenant>
    <name>ed_tenant_2</name>
    <id>87afae38ce1a498699463162ef919210</id>
    <authentication_source>Local</authentication_source>
    <status>Initialized</status>
    <sub_tenant_admin_list>
      <sub_tenant_admin>No Admin</sub_tenant_admin>
    </sub_tenant_admin_list>
  </sub_tenant>
  <sub_tenant>
    <name>ed_tenant_3</name>
    <id>38546e7ee00841eab1417125cee831e8</id>
    <authentication_source>Local</authentication_source>
    <status>Initialized</status>
    <sub_tenant_admin_list>
      <sub_tenant_admin>No Admin</sub_tenant_admin>
    </sub_tenant_admin_list>
  </sub_tenant>
  <sub_tenant>
    <name>maas_tenant</name>
    <id>c439ald5be734a8280d7003c5817fe45</id>
    <authentication_source>Local</authentication_source>
    <status>Initialized</status>
    <sub_tenant_admin_list>
      <sub_tenant_admin>No Admin</sub_tenant_admin>
    </sub_tenant_admin_list>
  </sub_tenant>
</sub_tenant_list>

```

## Modify a Subtenant

**Description** Lets you modify the default policy specification for the specified subtenant. Returns true if successful or an error message if unsuccessful. The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication. It must also include the `Accept` header with the value of `application/xml`.

|                           |  |
|---------------------------|--|
| <b>Required Role</b>      | TenantAdmin  |
| <b>HTTP Method</b>        | POST   |
| <b>URI</b>                | /tenant_admin/submit_change_default_spec   |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> <li>▷ <b>Accept:</b> Set to <code>application/xml</code>.</li> </ul> <p>HTTP body:</p> <ul style="list-style-type: none"> <li>▶ <b>default_spec_name:</b> Specify the name of the policy specification to set as the new default for the specified subtenant.</li> <li>▶ <b>sub_tenant_name:</b> Specify the name of the subtenant.</li> </ul> |
| <b>Request Header</b>     | <pre>POST /tenant_admin/submit_change_default_spec HTTP/1.1 Accept: application/xml Content-Type: application/x-www-form-urlencoded Cookie: _gui_session_id=df11382b45e740710b4a163260fcc407 Host: 10.5.116.110 Content-Length: 56</pre>   |
| <b>Request Body</b>       | <code>default_spec_name=testSpec&amp;sub_tenant_name=TestTenant</code>   |
| <b>Response Header</b>    | <pre>HTTP/1.1 200 OK Date: Mon, 14 Sep 2009 18:26:26 GMT Server: Mongrel 1.1.3 Status: 200 OK Cache-Control: no-cache Content-Type: application/xml; charset=utf-8 Content-Length: 24 Set-Cookie: _gui_session_id=df11382b45e740710b4a163260fcc407; path=/ Connection: close</pre>   |
| <b>Response Body</b>      | <p>A successful request returns:</p> <pre>&lt;cleared&gt;true&lt;/cleared&gt;</pre>  |

## Remove a Subtenant Admin

|                      |   |
|----------------------|---|
| <b>Description</b>   | Removes the specified user from the SubtenantAdmin role. Returns true if successful or an error message if unsuccessful. The HTTP request header must include the <code>Cookie</code> with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication. |
| <b>Required Role</b> | TenantAdmin   |

|                           |  |
|---------------------------|--|
| <b>HTTP Method</b>        | POST   |
| <b>URI</b>                | /tenant_admin/modify_sub_tenant_admin  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▶ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>HTTP body:</p> <ul style="list-style-type: none"> <li>▶ <b>operation:</b> Specify the operation type; for this request, the value is always <code>delete</code>.</li> <li>▶ <b>sub_tenant_name:</b> Specify the subtenant name.</li> <li>▶ <b>username:</b> Specify the name of the user to be removed from the SubTenantAdmin role for the specified subtenant.</li> </ul> |
| <b>Request Header</b>     | <pre>POST /tenant_admin/modify_sub_tenant_admin HTTP/1.1 Accept: application/xml Content-Type: application/x-www-form-urlencoded Cookie: _gui_session_id=559f6115e5ed705e0b40157116d42301 Host: 10.5.116.110 Content-Length: 67</pre>  |
| <b>Request Body</b>       | <code>operation=delete&amp;sub_tenant_name=TestTenant&amp;username=TestTemp</code>   |
| <b>Response Header</b>    | <pre>HTTP/1.1 200 OK Date: Mon, 14 Sep 2009 19:12:22 GMT Server: Mongrel 1.1.3 Status: 200 OK Cache-Control: no-cache Content-Type: application/xml; charset=utf-8 Content-Length: 24 Set-Cookie: _gui_session_id=559f6115e5ed705e0b40157116d42301; path=/ Connection: close</pre>   |
| <b>Response Body</b>      | <p>A successful request returns:</p> <pre>&lt;cleared&gt;true&lt;/cleared&gt;</pre>  |

---

## UID API Reference

- ▶ [Create UID](#)
- ▶ [Delete UID](#)
- ▶ [Disable UID](#)
- ▶ [Get UID](#)
- ▶ [List UIDs](#)

## Create UID

|                           |   |
|---------------------------|---|
| <b>Description</b>        | Creates a UID with the name specified in the <code>app_name</code> HTTP body parameter, then generates and returns a shared secret for it. The HTTP request header must include the <code>Cookie</code> with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication.   |
| <b>Required Role</b>      | TenantAdmin   |
| <b>HTTP Method</b>        | POST  |
| <b>URI</b>                | <code>/sub_tenant_admin/add_application</code>  |
| <b>Request Parameters</b> | HTTP header: <ul style="list-style-type: none"><li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li></ul> HTTP body: <ul style="list-style-type: none"><li>▷ <b>app_name:</b> Specify a string representing the UID. It must be unique.</li><li>▷ <b>sub_tenant_name:</b> Specify the name of the subtenant that owns the UID.</li><li>▷ <b>email:</b> Specify an email address.</li></ul> |
| <b>Request Header</b>     | <pre>POST /sub_tenant_admin/add_application HTTP/1.1 Accept: application/xml Content-Type: application/x-www-form-urlencoded Cookie: _gui_session_id=283ef56e4824548d162f9cfe273ae7ae Host: host:port Content-Length: 56</pre>  |
| <b>Request Body</b>       | <pre>app_name=samplenameUid0&amp;sub_tenant_name=samplename&amp;email=a@b.com</pre>   |
| <b>Response Header</b>    | <pre>HTTP/1.1 200 OK Date: Mon, 06 Jul 2009 17:40:22 Server: Mongrel 1.1.3 Status: 200 OK Cache-Control: no-cache Content-Type: application/xml; charset=utf-8 Content-Length: 60 Set-Cookie: _gui_session_id=283ef56e4824548d162f9cfe273ae7ae; path=/ Connection: close</pre>  |
| <b>Response Body</b>      | A successful request returns a shared secret:<br><pre>&lt;shared_secret&gt;MSuDvtqmivSS4qRsvHyrAMuyY8Y=&lt;/shared_secret&gt;</pre> An unsuccessful request returns an error message, for example:<br><pre>&lt;shared_secret&gt;Error: UID TestUid0 existing&lt;/shared_secret&gt;</pre>  |

## Delete UID

**Description** Deletes the specified UID (`app_name`) for the specified subtenant. The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role** TenantAdmin

**HTTP Method** GET

**URI** `/sub_tenant_admin/del_application`

**Request Parameters** HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

QueryString parameters:

- ▷ **app\_name:** Specify the application name (UID) you want to delete.
- ▷ **sub\_tenant\_name:** Specify the name of the subtenant that owns the UID.

**Request Header** GET /sub\_tenant\_admin/del\_application?app\_name=TestUid9&sub\_tenant\_name=Test  
HTTP/1.1  
Accept: application/xml  
Cookie: `_gui_session_id=cf93c65fbf89260f70f221936ed29c7e`  
Host: `host:port`

**Response Header** HTTP/1.1 200 OK  
Date: Mon, 06 Jul 2009 17:43:35 GMT  
Server: Mongrel 1.1.3  
Status: 200 OK  
Cache-Control: no-cache  
Content-Type: application/xml; charset=utf-8  
Set-Cookie: `_gui_session_id=cf93c65fbf89260f70f221936ed29c7e; path=/`  
Connection: close

**Response Body** A successful request returns:

```
<<shared_secret>+i/jVCzHz1pErmhMJ6ZECra2CFg= has been removed successfully.</shared_secret>
```

An unsuccessful request returns:

```
<shared_secret>Error: application TestUid9 doesn't existing</shared_secret>
```

or

```
<shared_secret>No such sub tenant or no permission to access requested sub tenant</shared_secret>
```

## Disable UID

|                           |   |
|---------------------------|---|
| <b>Description</b>        | Changes the state of the specified UID to disabled. Disabled UIDs cannot be used in data requests. You pass the UID and subtenant names in as parameters in the request body. The HTTP request header must include the <code>Cookie</code> with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication.                                  |
| <b>Required Role</b>      | TenantAdmin   |
| <b>HTTP Method</b>        | POST  |
| <b>URI</b>                | <code>/sub_tenant_admin/disable_application</code>  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"><li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li></ul> <p>HTTP body:</p> <ul style="list-style-type: none"><li>▷ <b>app_name:</b> Specify the UID to disable.</li><li>▷ <b>sub_tenant_name:</b> Specify the name of the subtenant that owns the UID.</li></ul> |
| <b>Request Header</b>     | <pre>POST /sub_tenant_admin/disable_application HTTP/1.1 Accept: application/xml Content-Type: application/x-www-form-urlencoded Cookie: _gui_session_id=cf26c9cc550128f6dcd07bd1deaae4a2 Host: host:port Content-Length: 42</pre>  |
| <b>Request Body</b>       | <pre>app_name=TestUid6&amp;sub_tenant_name=Test</pre>   |
| <b>Response Header</b>    | <pre>HTTP/1.1 200 OK Date: Mon, 06 Jul 2009 18:14:43 GMT Server: Mongrel 1.1.3 Status: 200 OK Cache-Control: no-cache Content-Type: application/xml; charset=utf-8 Content-Length: 24 Set-Cookie: _gui_session_id=cf26c9cc550128f6dcd07bd1deaae4a2; path=/ Connection: close</pre>  |
| <b>Response Body</b>      | <p>A successful request returns:</p> <pre>&lt;cleared&gt;true&lt;/cleared&gt;</pre> <p>If the UID does not exist, it returns:</p>   |

```

HTTP/1.1 302 Moved Temporarily
Date: Mon, 06 Jul 2009 20:17:40 GMT
Server: Mongrel 1.1.3
Status: 302 Found
Location: https://host:port/user/service_unavailable
Cache-Control: no-cache
Content-Type: text/html; charset=utf-8
Content-Length: 114
Set-Cookie: _gui_session_id=62079dccb6d5804cf4295141081da900; path=/
Connection: close

<html>
  <body>You are being
    <a href="http://10.6.143.43/user/service_unavailable">redirected</a>.
  </body>
</html>

```

If the subtenant does not exist, or a permissions error occurs, it returns:

```

HTTP/1.1 200 OK
Date: Mon, 06 Jul 2009 20:18:19 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 86
Set-Cookie: _gui_session_id=578378c13e3ae258d09e66cb514483e1; path=/
Connection: close

<cleared>No such sub tenant or no permission to access requested sub tenant</cleared>

```

## Enable UID

|                           |   |
|---------------------------|---|
| <b>Description</b>        | Changes the state of a UID from disabled to enabled. The HTTP request header must include the Cookie with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication.  |
| <b>Required Role</b>      | TenantAdmin   |
| <b>HTTP Method</b>        | POST  |
| <b>URI</b>                | <code>/sub_tenant_admin/enable_application</code>   |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>HTTP body:</p> <ul style="list-style-type: none"> <li>▷ <b>app_name:</b> Specify the UID to enable.</li> <li>▷ <b>sub_tenant_name:</b> Specify the name of the subtenant that owns the UID.</li> </ul> |

**Request Header**

```
POST /sub_tenant_admin/enable_application HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Cookie: _gui_session_id=50a9726ce3b0bef91cfafbf1732fd589
Host: host:port
Content-Length: 42
```

**Request Body**

```
app_name=TestUId7&sub_tenant_name=Test
```

**Response Header**

```
HTTP/1.1 200 OK
Date: Mon, 06 Jul 2009 18:15:37 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 24
Set-Cookie: _gui_session_id=50a9726ce3b0bef91cfafbf1732fd589; path=/
Connection: close
```

**Response Body**

A successful request returns:

```
<cleared>>true</cleared>
```

An unsuccessful request returns a message like this:

```
<cleared>No such sub tenant or no permission to access requested sub tenant</cleared>
```

## Get UID

**Description** Returns an XML document containing the UIDs and shared secrets for the application name (UID) and subtenant name included in the querystring parameter of the HTTP request header. The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role** TenantAdmin

**HTTP Method** GET

**URI** `/sub_tenant_admin/get_uid`

**Request Parameters** HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

Querystring parameters:

- ▷ **app\_name:** Specify the UID whose shared secret you want to return.
- ▷ **sub\_tenant\_name:** Specify the name of the subtenant that owns the UID.

**Request Header** GET /sub\_tenant\_admin/get\_uid?app\_name=TestUid0&sub\_tenant\_name=Test HTTP/1.1  
Accept: application/xml  
Cookie: \_gui\_session\_id=035da1f5bddb4a8704734c6658eef70a  
Host: 10.6.143.43

**Request Body** None

**Response Header** HTTP/1.1 200 OK  
Date: Mon, 06 Jul 2009 17:42:21 GMT  
Server: Mongrel 1.1.3  
Status: 200 OK  
Cache-Control: no-cache  
Content-Type: application/xml; charset=utf-8  
Content-Length: 165  
Set-Cookie: \_gui\_session\_id=035da1f5bddb4a8704734c6658eef70a; path=/  
Connection: close

**Response Body** A successful request returns:

```
<uid_secret>
  <uid>TestUid0</uid>
  <shared_secret>MSuDvtqmivSS4qRsvHyrAMuyY8Y=</shared_secret>
  <status>enabled</status>
  <email>a@b.com</email>
</uid_secret>
```

An unsuccessful request returns:

```
<error_message>Specified UID does not belong to this tenant</error_message>
```

or

```
<error_message>No such sub tenant or no permission to access requested sub
tenant</error_message>
```

## List UIDs

**Description** Returns an XML document that contains the following information about the UIDs for the specified tenant:

- ▶ The UID name.
- ▶ The UID shared secret.
- ▶ The UID status (enabled or disabled).
- ▶ The email address that is associated with the UID.

The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role** TenantAdmin

|                           |   |
|---------------------------|---|
| <b>HTTP Method</b>        | GET   |
| <b>URI</b>                | /sub_tenant_admin/list_uid  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>Querystring parameters:</p> <ul style="list-style-type: none"> <li>▶ <b>sub_tenant_name:</b> Specify the name of the subtenant whose UIDs you want to list.</li> </ul> |
| <b>Request Header</b>     | <pre>GET /sub_tenant_admin/list_uid?sub_tenant_name=TestTenant HTTP/1.1 Accept: application/xml Cookie: _gui_session_id=bdcdd918da8292c74272f91e806b4d7b Host: 10.5.116.110 sub_tenant_name: The name of the affected subtenant.</pre>  |
| <b>Request Body</b>       | None  |
| <b>Response Header</b>    | <pre>HTTP/1.1 200 OK Date: Mon, 14 Sep 2009 20:12:40 GMT Server: Mongrel 1.1.3 Status: 200 OK Cache-Control: no-cache Content-Type: application/xml; charset=utf-8 Content-Length: 694 Set-Cookie: _gui_session_id=bdcdd918da8292c74272f91e806b4d7b; path=/ Connection: close</pre>   |
| <b>Response Body</b>      | A successful request returns:   |

```

<uid_secret_list>
  <uid_secret>
    <uid>TestUidA</uid>
    <shared_secret>u3ZXmQztwkgbwD6fMIywx96cOE0=</shared_secret>
    <status>enabled</status>
    <email></email>
  </uid_secret>
  <uid_secret>
    <uid>TestUidB</uid>
    <shared_secret>5G3sIG7o1cdI00y+quML8d2n3o8=</shared_secret>
    <status>enabled</status>
    <email>a@b.com</email>
  </uid_secret>
  <uid_secret>
    <uid>user1</uid>
    <shared_secret></shared_secret>
    <status>enabled</status>
    <email></email>
  </uid_secret>
  <uid_secret>
    <uid>TestUid1</uid>
    <shared_secret>kJzLB9ha46mVfc3lhSnNs+dTYeU=</shared_secret>
    <status>enabled</status>
    <email></email>
  </uid_secret>
</uid_secret_list>

```

---

## Shared Secret API Reference

- ▶ [Reset the Shared Secret](#)

### Reset the Shared Secret

|                           |   |
|---------------------------|---|
| <b>Description</b>        | Generates a new shared secret for the specified <code>app_name</code> (UID). If the specified UID does not exist, it is created. The HTTP request header must include the <code>Cookie</code> parameter with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication.                   |
| <b>Required Role</b>      | TenantAdmin   |
| <b>HTTP Method</b>        | POST  |
| <b>URI</b>                | <code>/sub_tenant_admin/add_application</code>  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▶ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>HTTP body:</p> <ul style="list-style-type: none"> <li>▶ <b>app_name:</b> Specify the UID whose shared secret needs to be regenerated.</li> </ul> |

- ▷ **sub\_tenant\_name**: Specify the name of the subtenant that owns the UID.
- ▷ **regenerate**: Specify the value as *yes*.

**Request Header**

```
POST /sub_tenant_admin/add_application HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Cookie: _gui_session_id=c4fc5be4097b2a9d1fe4ac7714adfba8
Host: host:port
Content-Length: 57
```

**Request Body**

```
app_name=TestUid7&sub_tenant_name=Test&regenerate=yes
```

**Response Header**

```
HTTP/1.1 200 OK
Date: Mon, 06 Jul 2009 18:16:32 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 60
Set-Cookie: _gui_session_id=c4fc5be4097b2a9d1fe4ac7714adfba8; path=/
Connection: close
```

**Response Body**

A successful request returns:

```
<shared_secret>7KVz1hlcyat9WnwrTafPyQxROZo=</shared_secret>
```

An unsuccessful request returns:

```
<shared_secret>No such sub tenant or no permission to access requested sub
tenant</shared_secret>
```

# 4 Working with CIFS Shares

This section describes how to programmatically export CIFS Atmos nodes. It includes the following topics:

- ▶ [Working with CIFS Shares](#)
- ▶ [API Reference](#)

All of the API calls require an authenticated session. To create an authenticated session, see [Chapter , “Authentication” on page 9](#).

---

## Working with CIFS Shares

This section describes the set of commands you use for:

- ▶ [Create a CIFS Node](#)
- ▶ [Create a CIFS share](#)
- ▶ [List all CIFS Shares for a Tenant/Subtenant](#)

### Create a CIFS Node

To add a CIFS node:

- 1 Authenticate as a tenant administrator. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).
- 2 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 3 If you do not have a UUID, convert the node name to its UUID. For more information, see [“Translate Node Name to UUID” on page 58](#).
- 4 Add the CIFS node using the operation described in [“Add CIFS Node” on page 48](#).

### Create a CIFS share

To create a CIFS share:

- 1 Authenticate as a tenant administrator. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).

- 2 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 3 If you do not have a UUID, convert the node name to its UUID. For more information, see [“Translate Node Name to UUID” on page 58](#).
- 4 If one does not already exist, add a CIFS node. For more information, see [“Add CIFS Node” on page 48](#).
- 5 Add the CIFS share using the operation described in [“Create CIFS Share” on page 52](#).

## List all CIFS Shares for a Tenant/Subtenant

To obtain the list of subtenants for a TenantAdmin:

- 1 Authenticate as an Atmos TenantAdmin. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).
- 1 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 2 If you do not have a UUID, convert the node name to its UUID. For more information, see [“Translate Node Name to UUID” on page 58](#).
- 2 Use the HTTP POST command with this URI:

```
/sub_tenant_admin/node_cifs_list
```

The following body shows an example of what is returned:

```
<result>
  <access_type>cifs</access_type>
  <cifs_node_list>
    <cifs_node>
      <share_name>abcd</share_name>
      <path>/mnt/mauifs/xyz1</path>
    </cifs_node>
    <cifs_node>
      <share_name>efgh</share_name>
      <path>/mnt/mauifs/efgh</path>
    </cifs_node>
  </cifs_node_list>
</result>
```

---

## API Reference

- ▶ [Add CIFS Node](#)
- ▶ [Change CIFS Share](#)
- ▶ [Create CIFS Share](#)
- ▶ [Delete CIFS Shares](#)
- ▶ [Global Configuration](#)
- ▶ [List CIFS Shares](#)
- ▶ [Translate Node Name to UUID](#)

### Add CIFS Node

|                           |  |
|---------------------------|--|
| <b>Description</b>        | Adds a node as a CIFS share to the specified tenant. Returns true if the request succeeds; otherwise returns an error message.   |
| <b>Required Role</b>      | TenantAdmin  |
| <b>HTTP Method</b>        | POST   |
| <b>URI</b>                | <code>/tenant_admin/submit_add_node_cifs</code>  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"><li>▶ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li></ul> <p>HTTP body:</p> <ul style="list-style-type: none"><li>▶ <b>node_uuid:</b> Specify the UUID of the CIFS node.</li><li>▶ <b>public:</b> Valid values:<ul style="list-style-type: none"><li>▶ <b>yes</b> — Specify when no password is required to connect to the service; privileges are those of the guest account. This is the default. This has the same meaning as the <code>guest_ok</code> parameter.</li><li>▶ <b>no</b> — Specify when a password is required to connect to the service.</li></ul></li><li>▶ <b>share_advance_configure:</b> Boolean value (Yes/No) that specifies whether the following parameters are used: <code>guest_ok</code>, <code>only_guest</code>, <code>valid_users</code>, <code>create_mode</code>, <code>directory_mode</code>, and <code>write_list</code>.)</li><li>▶ <b>read_only:</b> Valid values:<ul style="list-style-type: none"><li>▶ <b>yes</b> — Specify when users of the service cannot create or modify files in the service's directory.</li></ul></li></ul> |

- ▷ **no** — Specify when users of the service are allowed to create or modify files in the service’s directory. This is the default.
- ▶ **guest\_ok**: Do not specify this; it has the same meaning as **Public**, but only **Public** takes effect.
- ▶ **directory\_mode**: An octal value such as 0755 (the default) used when converting DOS modes to UNIX modes when creating UNIX directories. When a directory is created, the necessary permissions are calculated according to the mapping from DOS modes to UNIX permissions, and the resulting UNIX mode is bit-wise AND-ed with this parameter. This parameter may be thought of as a bit-wise mask for a directory’s UNIX access modes. Any bit not set here is removed from the modes set on a directory when it is created. The default value of this parameter removes the “group” and “other” write bits from the UNIX mode, allowing only the user who owns the directory to modify it.
- ▶ **share\_name**: Specifies the directory name shown in the Windows environment after CIFS sharing.
- ▶ **valid\_users**: Specifies the set of users (a comma separated list) who are allowed to login to this service. If this is empty (the default), any user can login. Every name represents a SAMBA/CIFS user.
- ▶ **browseable**: Specifies whether this share is shown in the list of available shares in a net view and in the browse list. Values are: yes/no. The default is yes.
- ▶ **write\_list**: Specifies the set of users (a comma separated list) who have read-write access to a service. This option overrides the **Read Only** parameter.
- ▶ **path**: The root directory of the CIFS access on the specified node. This is a directory to which the user of the service is to be given access. This should be the full path name of the directory from the root of the tenant namespace.
- ▶ **comment**: A comment is required in the CIFS configuration file. This text field is seen next to a share when a client queries the server, either via the network neighborhood or via net view to list what shares are available.
- ▶ **only\_guest**: If Yes, only guest connections to the service are permitted. This parameter has no effect unless **Public** is set for the service. The default is Yes.
- ▶ **only\_guest**: If Yes, only guest connections to the service are permitted. This parameter has no effect unless **Public** is set for the service. The default is Yes.
- ▶ **create\_mode**: Specify an octal value such as 0744 (the default). Used when a file is created, the necessary permissions are calculated according to the mapping from DOS modes to UNIX permissions, and the resulting UNIX mode is bit-wise AND-ed with this parameter. This parameter may be thought of as a bit-wise mask for a file’s UNIX access modes. Any bit not set here is removed from the modes set on a file when it is created. The default value of this parameter removes the “group” and “other” write and execute bits from the UNIX modes.

## Request Header

```
POST /tenant_admin/submit_add_node_cifs HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Cookie: _gui_session_id=1c05b07ab4c4e08fa3d91f6b39035f86
Host: 10.5.116.110
Content-Length: 257
```

**Request Body** node\_uuid=564DE0F3-A5BF-9DFD-8C5B-5EA182150521&public=yes&share\_advance\_configure=disable&read\_only=yes&guest\_ok=yes&directory\_mode=&share\_name=Test9&valid\_users=&browsable=yes&write\_list=&path=Test10&comment=this+is+a+test+cifs&only\_guest=yes&create\_mode=

**Response Header** HTTP/1.1 200 OK  
Date: Fri, 24 Jul 2009 18:54:40 GMT  
Server: Mongrel 1.1.3  
Status: 200 OK  
Cache-Control: no-cache  
Content-Type: application/xml; charset=utf-8  
Content-Length: 20  
Set-Cookie: \_gui\_session\_id=1c05b07ab4c4e08fa3d91f6b39035f86; path=/  
Connection: close

**Response Body** A successful request returns the following:

```
<added>>true</added>
```

A failed request returns a message like the following:

```
<added>Add node cifs error.</added>
```

## Change CIFS Share

**Description** Lets you modify the properties of an existing CIFS share.

**Required Role** TenantAdmin

**HTTP Method** POST

**URI** /sub\_tenant\_admin/submit\_configure\_node\_cifs

**Request Parameters** HTTP Header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

HTTP Body:

- ▶ **node\_uuid:** Specify the UUID of the CIFS node.
- ▶ **subtenant\_name:** Specify the name of the subtenant that owns the CIFS node.
- ▶ **public:** Valid values:
  - ▷ **yes** — Specify when no password is required to connect to the service; privileges are those of the guest account. This is the default. This has the same meaning as the `guest_ok` parameter.
  - ▷ **no** — Specify when a password is required to connect to the service.
- ▶ **share\_advance\_configure:** Boolean value (Yes/No) that specifies whether the following parameters are used: `guest_ok`, `only_guest`, `valid_users`, `create_mode`, `directory_mode`, and `write_list`.)

- ▶ **read\_only:** Valid values:
  - ▷ **yes** — Specify when users of the service cannot create or modify files in the service’s directory.
  - ▷ **no** — Specify when users of the service are allowed to create or modify files in the service’s directory. This is the default.
- ▶ **guest\_ok:** Do not specify this; it has the same meaning as **Public**, but only **Public** takes effect.
- ▶ **directory\_mode:** An octal value such as 0755 (the default) used when converting DOS modes to UNIX modes when creating UNIX directories. When a directory is created, the necessary permissions are calculated according to the mapping from DOS modes to UNIX permissions, and the resulting UNIX mode is bit-wise AND-ed with this parameter. This parameter may be thought of as a bit-wise mask for a directory’s UNIX access modes. Any bit not set here is removed from the modes set on a directory when it is created. The default value of this parameter removes the “group” and “other” write bits from the UNIX mode, allowing only the user who owns the directory to modify it.
- ▶ **share\_name:** Specifies the directory name shown in the Windows environment after CIFS sharing.
- ▶ **valid\_users:** Specifies the set of users (a comma separated list) who are allowed to login to this service. If this is empty (the default), any user can login. Every name represents a SAMBA/CIFS user.
- ▶ **browseable:** Specifies whether this share is shown in the list of available shares in a net view and in the browse list. Values are: yes/no. The default is yes.
- ▶ **write\_list:** Specifies the set of users (a comma separated list) who have read-write access to a service. This option overrides the **Read Only** parameter.
- ▶ **path:** The root directory of the CIFS access on the specified node. This is a directory to which the user of the service is to be given access. This should be the full path name of the directory from the root of the tenant namespace.
- ▶ **comment:** A comment is required in the CIFS configuration file. This text field is seen next to a share when a client queries the server, either via the network neighborhood or via net view to list what shares are available.
- ▶ **only\_guest:** If Yes, only guest connections to the service are permitted. This parameter has no effect unless **Public** is set for the service. The default is Yes.
- ▶ **create\_mode:** Specify an octal value such as 0744 (the default). Used when a file is created, the necessary permissions are calculated according to the mapping from DOS modes to UNIX permissions, and the resulting UNIX mode is bit-wise AND-ed with this parameter. This parameter may be thought of as a bit-wise mask for a file’s UNIX access modes. Any bit not set here is removed from the modes set on a file when it is created. The default value of this parameter removes the “group” and “other” write and execute bits from the UNIX modes.

## Request Header

```
POST /sub_tenant_admin/submit_configure_node_cifs HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Cookie: _gui_session_id=2267c906b87b35b46cef910be1790599
Host: 10.5.116.244
Content-Length: 282
```

**Request Body** node\_uuid=564D1DDC-8B85-5B06-BA35-F0A26889C6B6&subtenant\_name=t1&public=no&share\_advance\_configure=disable&read\_only=no&guest\_ok=yes&directory\_mode=&share\_name=abcd&valid\_users=&browseable=no&write\_list=&path=abcd&comment=test+comment&only\_guest=no&create\_mode=

**Response Header** HTTP/1.1 200 OK  
Connection: close  
Date: Wed, 20 Jan 2010 09:49:26 GMT  
Set-Cookie: \_gui\_session\_id=2267c906b87b35b46cef910be1790599; path=/  
Status: 200 OK  
X-Runtime: 3358ms  
ETag: "27430fa0a0a46a8af142853e782c1a44"  
Cache-Control: private, max-age=0, must-revalidate  
Server: Mongrel 1.1.5  
Content-Type: application/xml; charset=utf-8  
Content-Length: 24

**Response Body** <cleared>true</cleared>

## Create CIFS Share

**Description** Creates a CIFS share for the specified node. Returns true if successful.

**Required Role** TenantAdmin

**HTTP Method** POST

**URI** /sub\_tenant\_admin/submit\_add\_node\_cifs

**Request Parameters** HTTP Header:

- ▶ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

HTTP Body:

- ▶ **node\_uuid:** Specify the UUID of the CIFS node.
- ▶ **subtenant\_name:** Specify the name of the subtenant that owns the CIFS node.
- ▶ **public:** Valid values:
  - ▷ **yes** — Specify when no password is required to connect to the service; privileges are those of the guest account. This is the default. This has the same meaning as the `guest_ok` parameter.
  - ▷ **no** — Specify when a password is required to connect to the service.
- ▶ **share\_advance\_configure:** Boolean value (Yes/No) that specifies whether the following parameters are used: `guest_ok`, `only_guest`, `valid_users`, `create_mode`, `directory_mode`, and `write_list`.
- ▶ **read\_only:** Valid values:
  - ▷ **yes** — Specify when users of the service cannot create or modify files in the service's directory.

- ▷ **no** — Specify when users of the service are allowed to create or modify files in the service’s directory. This is the default.
- ▶ **guest\_ok**: Do not specify this; it has the same meaning as **Public**, but only **Public** takes effect.
- ▶ **directory\_mode**:
- ▶ **share\_name**: Specifies the directory name shown in the Windows environment after CIFS sharing.
- ▶ **valid\_users**: A list of users allowed to login. If this is empty, then any user can login.
- ▶ **browseable**: Specifies whether this share is shown in the list of available shares in a net view and in the browse list. Values are: yes/no. The default is yes.
- ▶ **write\_list**: Specifies the set of users (a comma separated list) who have read-write access to a service. This option overrides the **Read Only** parameter.
- ▶ **path**: The root directory of the CIFS access on the specified node. This is a directory to which the user of the service is to be given access. This should be the full path name of the directory from the root of the tenant namespace.
- ▶ **comment**: A comment is required in the CIFS configuration file. This text field is seen next to a share when a client queries the server, either via the network neighborhood or via net view to list what shares are available.
- ▶ **only\_guest**: If Yes, only guest connections to the service are permitted. This parameter has no effect unless **Public** is set for the service. The default is Yes.
- ▶ **create\_mode**: Specify an octal value such as 0744 (the default). Used when a file is created, the necessary permissions are calculated according to the mapping from DOS modes to UNIX permissions, and the resulting UNIX mode is bit-wise AND-ed with this parameter. This parameter may be thought of as a bit-wise mask for a file’s UNIX access modes. Any bit not set here is removed from the modes set on a file when it is created. The default value of this parameter removes the “group” and “other” write and execute bits from the UNIX modes.

**Request Header**

```
POST /sub_tenant_admin/submit_add_node_cifs HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Cookie: _gui_session_id=b8e72170a0774dae8f359e347e9af942
Host: 10.5.116.244
Content-Length: 282
```

**Request Body**

```
node_uuid=564D1DDC-8B85-5B06-BA35-F0A26889C6B6&subtenant_name=t1&public=no&share_advance_configure=disable&read_only=no&guest_ok=yes&directory_mode=&share_name=xyz&valid_users=&browseable=no&write_list=&path=xyz&comment=test+comment&only_guest=no&create_mode=
```

**Response Header**  
HTTP/1.1 200 OK  
Connection: close  
Date: Wed, 20 Jan 2010 10:21:57 GMT  
Set-Cookie: \_gui\_session\_id=b8e72170a0774dae8f359e347e9af942; path=/  
Status: 200 OK  
X-Runtime: 6560ms  
ETag: "b98201b5ee2afcc512464a9c0842a7be"  
Cache-Control: private, max-age=0, must-revalidate  
Server: Mongrel 1.1.5  
Content-Type: application/xml; charset=utf-8  
Content-Length: 20

**Response Body**  
<added>true</added>

## Delete CIFS Shares

**Description** Deletes the specified CIFS share.

**Required Role** TenantAdmin

**HTTP Method** POST

**URI** /sub\_tenant\_admin/submit\_delete\_node\_cifs

**Request Parameters** HTTP Header:

- ▶ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

HTTP Body:

- ▶ **node\_uuid:** Specify the UUID of the CIFS node.
- ▶ **subtenant\_name:** Specify the name of the subtenant that owns the CIFS node.
- ▶ **share\_name:** Specifies the directory name shown in the Windows environment after CIFS sharing.

**Request Header**  
POST /sub\_tenant\_admin/submit\_delete\_node\_cifs HTTP/1.1  
Accept: application/xml  
Content-Type: application/x-www-form-urlencoded  
Cookie: \_gui\_session\_id=d076332976f4c44c7caedb881241d10e  
Host: 10.5.116.244  
Content-Length: 80

**Request Body**  
node\_uuid=564D1DDC-8B85-5B06-BA35-F0A26889C6B6&subtenant\_name=t1&share\_name=test

**Response Header**  
HTTP/1.1 200 OK  
Connection: close  
Date: Wed, 20 Jan 2010 08:37:45 GMT  
Set-Cookie: \_gui\_session\_id=d076332976f4c44c7caedb881241d10e; path=/  
Status: 200 OK  
X-Runtime: 1324ms  
ETag: "0d7da07534bfa39e841b8ee8cba2b4b2"  
Cache-Control: private, max-age=0, must-revalidate  
Server: Mongrel 1.1.5  
Content-Type: application/xml; charset=utf-8  
Content-Length: 24

**Response Body**  
<deleted>true</deleted>

## Global Configuration

**Description** Lets you define the global settings for all CIFS shares on the specified node.

**Required Role** TenantAdmin

**HTTP Method** POST

**URI** /sub\_tenant\_admin/submit\_configure\_node\_global\_cifs

**Request Parameters** HTTP Header:

- ▶ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

HTTP Body:

- ▶ **workgroup:** The workgroup your server will appear to be in when queried by clients. The default is WORKGROUP. (This parameter also controls the Domain name used with the Security = Domain setting, which currently is not supported.) When Security is set to ADS, Workgroup is the short name of the domain; commonly this is the same as the first part of the Realm name. If you define the wrong value, the server will not be able to join the ADS domain.
- ▶ **server\_string:** The string that will appear in the browse lists next to the machine name. The default is Samba Server.
- ▶ **netbios\_name:** The NetBIOS name by which a server is known. By default, this is the same as the first component of the host's DNS name.
- ▶ **host\_allow:** the set of hosts which are permitted to access a service. Settings apply to all services, regardless of whether the individual service has a different setting. You can specify the hosts by name or IP number. You also can specify hosts by network/netmask pairs and by netgroup names if your system supports netgroups. The default is no entry, meaning all hosts are allowed.
- ▶ **security:** The Security option affects how clients respond to the server. There are two options, Share and ADS. (There are two security modes Atmos does not support: User (because Atmos does not support internal CIFS user management) and Domain.)

- ▶ **realm:** The fully qualified domain name of the database server. The realm is used as the Active Directory service equivalent of the NT4 domain. This must be specified in all-capital letters.
- ▶ **preferred\_master:** Controls the election of a master node in the CIFS domain and whether nmbd is a preferred master browser for its workgroup. (nmbd is a daemon service involved in the CIFS server. It queries the domain or share names in the domain.) Valid values are Yes, No, and Auto. If Yes, nmbd has a slight advantage in winning the election. If No, nmbd cannot win the election. If Auto, the election is random, this is the default
- ▶ **encrypt\_passwords:** Specifies whether to encrypt CIFS user passwords. By default, Windows NT 4.0 SP3 and above and Windows 98 expect encrypted passwords. The default is Yes. Default: encrypt passwords = yes
- ▶ **winbind\_use\_default\_domain:** Specifies whether the winbindd daemon operates on users without a domain component in their usernames. Users without a domain component are treated as part of the winbindd server's domain. The default is No. Default: winbind use default domain = no. Example: winbind use default domain = yes
- ▶ **idmap\_uid:** 12345678-11223344. Specifies the range of user IDs that are allocated for the purpose of mapping UNIX users to NT user SIDs. This range of user IDs should not contain any existing local or NIS users; i.e., it should be beyond the maximum UID value of the local system, to avoid overlapping. A sample range is 10000-20000. Default: idmap gid = Example: idmap gid = 10000-20000
- ▶ **idmap\_gid:** 12345678-11223344. Specifies the range of group IDs that are allocated for the purpose of mapping UNIX groups to NT group SIDs. This range of group IDs should not contain any existing local or NIS groups; i.e., it should be beyond the maximum GID value of the local system, to avoid overlapping. A sample range is 10000 - 20000. Default: idmap gid = Example: idmap gid = 10000-20000
- ▶ **ad\_server:** A directory service used to store information about the network resources. The Active Directory server is either an IP address or a simple host name.
- ▶ **ad\_admin:** The user name of the Active Directory administrator.
- ▶ **ad\_password:** The password of the Active Directory administrator.
- ▶ **node\_uuid:** Specify the UUID of the CIFS node.
- ▶ **subtenant\_name:** Specify the name of the subtenant that owns the CIFS node.
- ▶ **share\_name:** Specify the name of the CIFS share.

**Request Header**

```
POST /sub_tenant_admin/submit_configure_node_global_cifs HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Cookie: _gui_session_id=5e2d48c06e3a30f39ef0e1e18b1703a0
Host: 10.5.116.244
Content-Length: 326
```

**Request Body**

```
workgroup=WORKGROUP&server_string=samba&netbios_name=netbios&host_allow=168.158.116.128&security=share&realm=&preferred_master=no&encrypt_passwords=no&winbind_use_default_domain=no&idmap_uid=12345678-11223344&idmap_gid=12345678-11223344&ad_server=&ad_admin=&ad_password=&node_uuid=564D1DDC-8B85-5B06-BA35-F0A26889C6B6&subtenant_name=t1
```

**Response Header**  
HTTP/1.1 200 OK  
Connection: close  
Date: Wed, 20 Jan 2010 10:48:30 GMT  
Set-Cookie: \_gui\_session\_id=5e2d48c06e3a30f39ef0e1e18b1703a0; path=/  
Status: 200 OK  
X-Runtime: 1853ms  
ETag: "c24c30bf2d4618274b40014eeeb462c9"  
Cache-Control: private, max-age=0, must-revalidate  
Server: Mongrel 1.1.5  
Content-Type: application/xml; charset=utf-8  
Content-Length: 54

**Response Body**  
<result>Global cifs configured successfully.</result>

## List CIFS Shares

**Description** Returns the list of CIFS shares configured for a specified node.

**Required Role** TenantAdmin

**HTTP Method** POST

**URI** /sub\_tenant\_admin/node\_cifs\_list

**Request Parameters** HTTP Header:

- ▶ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

HTTP Body:

- ▶ **node\_uuid:** Specify the UUID of the CIFS node.
- ▶ **subtenant\_name:** Specify the name of the subtenant that owns the CIFS node.

**Request Header**  
POST /sub\_tenant\_admin/node\_cifs\_list HTTP/1.1  
Accept: application/xml  
Content-Type: application/x-www-form-urlencoded  
Cookie: \_gui\_session\_id=e0c4c74cbe985e0c9447f4b976b23e21  
Host: 10.5.116.244  
Content-Length: 64

**Request Body**  
node\_uuid=564D1DDC-8B85-5B06-BA35-F0A26889C6B6&subtenant\_name=t1

**Response Header**

```
HTTP/1.1 200 OK
Connection: close
Date: Wed, 20 Jan 2010 08:12:10 GMT
Set-Cookie: _gui_session_id=e0c4c74cbe985e0c9447f4b976b23e21; path=/
Status: 200 OK
X-Runtime: 1512ms
ETag: "1b6a9c4eff7f42880fe224256c25ff01"
Cache-Control: private, max-age=0, must-revalidate
Server: Mongrel 1.1.5
Content-Type: application/xml; charset=utf-8
Content-Length: 202
```

**Response Body**

```
<result>
  <access_type>cifs</access_type>
  <cifs_node_list>
    <cifs_node>
      <share_name>test</share_name>
      <path>/mnt/mauifs/sharepath</path>
    </cifs_node>
  </cifs_node_list>
</result>
```

## Translate Node Name to UUID

**Description** Returns the translated UUID of the Node name you pass in.

**Required Role** TenantAdmin

**HTTP Method** GET

**URI** /util/translate

**Request Parameters** HTTP header:

- ▶ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

Querystring parameters:

- ▶ **scope:** Specify the type of conversion to perform on the specified value. For this request, the value is always `Node`.
- ▶ **input:** Specify the input type for the specified value. For this request, the value is always `name`.
- ▶ **value:** Specify the node name to convert.
- ▶ **output:** Specify the output type for the specified value. For this request, always specify `uuid`.

**Request Header**

```
GET /util/translate?scope=Node&input=name&value=Testtest1-001&output=uuid HTTP/1.1
Accept: application/xml
Cookie: _gui_session_id=1c05b07ab4c4e08fa3d91f6b39035f86
Host: 10.5.116.110
```

**Response Header**

```
HTTP/1.1 200 OK
Date: Fri, 24 Jul 2009 18:54:40 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 68
Set-Cookie: _gui_session_id=1c05b07ab4c4e08fa3d91f6b39035f86; path=/
Connection: close
```

**Response Body**

A successful request returns the node ID.

```
<translated_id>564DE0F3-A5BF-9DFD-8C5B-5EA182150521<translated_id>
```

An unsuccessful request returns the following:

```
<translated_id><translated_id>
```

# 5 Working with NFS Shares

This section describes how to programmatically export NFS Atmos nodes. It includes the following topics:

- ▶ [Working with NFS Shares](#)
- ▶ [API Reference](#)

All of the API calls require an authenticated session. To create an authenticated session, see [Chapter , “Authentication” on page 9](#).

---

## Working with NFS Shares

This section describes the set of commands you use for:

- ▶ [Adding an NFS Node](#)
- ▶ [Creating an NFS Share](#)
- ▶ [Listing the NFS Shares for a Tenant/Subtenant](#)

### Adding an NFS Node

- 1 Authenticate as a tenant administrator. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).
- 2 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 3 If you do not have a UUID, convert the node name to its UUID. For more information, see [“Create NFS Node” on page 62](#).
- 4 Add the NFS node. For more information, see [“Create NFS Node” on page 62](#).

### Creating an NFS Share

Follow these steps, to create an NFS share:

- 1 Authenticate as a tenant administrator. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).
- 2 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.

- 3 If you do not have a UUID, convert the node name to its UUID. For more information, see [“Create NFS Node” on page 62](#).
- 4 If one does not already exist, add a NFS node. For more information, see [“Create NFS Node” on page 62](#).
- 5 Add the NFS share. For more information, see [“Create NFS Shares” on page 64](#).

## Listing the NFS Shares for a Tenant/Subtenant

- 1 Authenticate as a tenant administrator. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).
- 2 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 3 If you do not have a UUID, convert the node name to its UUID. For more information, see [“Create NFS Node” on page 62](#).
- 4 Use the HTTP POST verb with the following URI:

```
/sub_tenant_admin/node_nfs_list
```

The following shows an example of what is returned:

```
<result>
  <access_type>nfs</access_type>
  <nfs_node_list>
    <nfs_node>
      <share_path>/mnt/mauifs/test</share_path>
      <host>test1</host>
    </nfs_node>
    <nfs_node>
      <share_path>/mnt/mauifs/abcd</share_path>
      <host>host1</host>
    </nfs_node>
    <nfs_node>
      <share_path>/mnt/mauifs/efgh</share_path>
      <host>test1</host>
    </nfs_node>
  </nfs_node_list>
</result>
```

---

## API Reference

- ▶ [Create NFS Node](#)
- ▶ [Create NFS Shares](#)
- ▶ [Change NFS Shares](#)
- ▶ [Delete NFS Shares](#)

- ▶ [List NFS Shares](#)
- ▶ [Create NFS Node](#)

## Create NFS Node

|                           |   |
|---------------------------|---|
| <b>Description</b>        | Adds a node as an NFS share to the specified tenant. Returns true if the request succeeds; otherwise returns an error message.  |
| <b>Required Role</b>      | TenantAdmin   |
| <b>HTTP Method</b>        | POST  |
| <b>URI</b>                | /tenant_admin/submit_add_node_nfs   |
| <b>Request parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▶ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>HTTP Body:</p> <ul style="list-style-type: none"> <li>▶ <b>node_uuid:</b> The UUID of the NFS node to add.</li> <li>▶ <b>share_path:</b> The root directory of the NFS access on the specified node. This is a directory to which the user of the service is to be given access. This should be the full path name of the directory from the root of the tenant namespace. <ul style="list-style-type: none"> <li><i>This path name must include the subtenant directory name, if you specified Multi Subtenant Access when you added this node to a tenant. For example, if the directory to be shared is NFS_Share_A and is within the namespace of a subtenant with ID ad67a845aec4421badc97aabf66b4b08, the share path should be entered as ad67a845aec4421badc97aabf66b4b08/NFS_Share_A</i></li> </ul> </li> <li>▶ <b>host:</b> The hosts that access the share. Valid values are: <ul style="list-style-type: none"> <li>▷ <i>Single machine</i> — A fully qualified domain name (which can be resolved by the server), hostname (which can be resolved by the server), or IP address.</li> <li>▷ <i>Series of machines specified with wildcards</i> — Use the * or ? character to specify a string match. Wildcards are not to be used with IP addresses; however, they may work accidentally if reverse DNS lookups fail. When specifying wildcards in fully qualified domain names, dots (.) are not included in the wildcard. For example, *.example.com includes one.example.com but does not include one.two.example.com.</li> <li>▷ <i>IP networks</i> — Use a.b.c.d/z, where a.b.c.d is the network and z is the number of bits in the netmask (for example, 192.168.0.0/24). Another acceptable format is a.b.c.d/netmask, where a.b.c.d is the network and netmask is the netmask (for example, 192.168.100.8/255.255.255.0).</li> <li>▷ <i>Netgroups</i> — Use the format @group-name, where group-name is the NIS netgroup name.</li> </ul> </li> <li>▶ <b>io:</b> Specify r or rw. Specifies whether the directory is read-only (r) or has read/write (rw) permissions. The default is rw.</li> </ul> |

- ▶ **squash:** User access. Valid values are:
  - ▷ *all\_squash* — Treat all client users as anonymous users.
  - ▷ *root\_squash* — Do not treat a remote root user as local root.
  - ▷ *no\_root\_squash* — Treat a remote root user as local root. This is the default.
- ▶ **sync:** Values are: yes or no. When yes, the server cannot reply to requests until the changes made by the request are written to the disk. The default value is yes.
- ▶ **idmap:** Map remote id to local one, which can be used to improve the NFS security, for example, with *root\_squash*, remote root will be mapped to local *anonuid/anongid*.
- ▶ **secure:** Values are yes or no. If yes, all requests must originate from a port lower than 1024. If no, requests from any port number are accepted. The default is yes.
- ▶ **anonuid:** A numeric value that represents the anonymous UID. Sets the user ID of the anonymous account, to map all requests to one user.
- ▶ **anongid:** A numeric value that represents the anonymous GID. Sets the group ID of the anonymous account, to map all requests to one user.
- ▶ **squash\_uids:** A list of user IDs that are subject to anonymous mapping; for example: 0-15,20,25-50.
- ▶ **squash\_gids:** A list of group IDs that are subject to anonymous mapping; for example, 0-15,20,25-50.

**Request Header**

```
POST /tenant_admin/submit_add_node_nfs HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Cookie: _gui_session_id=404bfff1db3641996f8ee61860293ff7
Host: 10.6.143.43
Content-Length: 180
```

**Request Body**

```
node_uuid=564D3B55-E772-02C8-4D9F-26A16DDD8E14&share_path=/srv/sysmgmt&host=10.5.115.145&io=rw&squash=root_squash&sync=n&idmap=&secure=y&anonuid=&anongid=&squash_uids=&squash_gids=
```

**Response Header**

```
HTTP/1.1 200 OK
Date: Tue, 14 Jul 2009 19:13:34 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 20
Set-Cookie: _gui_session_id=404bfff1db3641996f8ee61860293ff7; path=/
Connection: close
```

**Response Body**

A successful request returns:

```
<added>>true</added>
```

A failed request returns a message like:

```
<added>Add node nfs entry error</added>
```

## Create NFS Shares

|                           |  |
|---------------------------|--|
| <b>Description</b>        | Adds an NFS share to an existing NFS node.   |
| <b>Required Role</b>      | TenantAdmin  |
| <b>HTTP Method</b>        | POST   |
| <b>URI</b>                | /sub_tenant_admin/submit_add_node_nfs  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"><li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li></ul> <p>HTTP Body:</p> <ul style="list-style-type: none"><li>▶ <b>node_uuid:</b> The UUID of the NFS node to add.</li><li>▶ <b>subtenant_name:</b></li><li>▶ <b>share_path:</b> The root directory of the NFS access on the specified node. This is a directory to which the user of the service is to be given access. This should be the full path name of the directory from the root of the tenant namespace.<br/><br/><i>This path name must include the subtenant directory name, if you specified Multi Subtenant Access when you added this node to a tenant. For example, if the directory to be shared is <code>NFS_Share_A</code> and is within the namespace of a subtenant with ID <code>ad67a845aec4421badc97aabf66b4b08</code>, the share path should be entered as <code>ad67a845aec4421badc97aabf66b4b08/NFS_Share_A</code></i></li><li>▶ <b>host:</b> The hosts that access the share. Valid values are:<ul style="list-style-type: none"><li>▷ <i>Single machine</i> — A fully qualified domain name (which can be resolved by the server), hostname (which can be resolved by the server), or IP address.</li><li>▷ <i>Series of machines specified with wildcards</i> — Use the <code>*</code> or <code>?</code> character to specify a string match. Wildcards are not to be used with IP addresses; however, they may work accidentally if reverse DNS lookups fail. When specifying wildcards in fully qualified domain names, dots (<code>.</code>) are not included in the wildcard. For example, <code>*.example.com</code> includes <code>one.example.com</code> but does not include <code>one.two.example.com</code>.</li><li>▷ <i>IP networks</i> — Use <code>a.b.c.d/z</code>, where <code>a.b.c.d</code> is the network and <code>z</code> is the number of bits in the netmask (for example, <code>192.168.0.0/24</code>). Another acceptable format is <code>a.b.c.d/netmask</code>, where <code>a.b.c.d</code> is the network and <code>netmask</code> is the netmask (for example, <code>192.168.100.8/255.255.255.0</code>).</li><li>▷ <i>Netgroups</i> — Use the format <code>@group-name</code>, where <code>group-name</code> is the NIS netgroup name.</li></ul></li><li>▶ <b>io:</b> Specify <code>r</code> or <code>rw</code>. Specifies whether the directory is read-only (<code>r</code>) or has read/write (<code>rw</code>) permissions. The default is <code>rw</code>.</li><li>▶ <b>squash:</b> User access. Valid values are:<ul style="list-style-type: none"><li>▷ <i>all_squash</i> — Treat all client users as anonymous users.</li></ul></li></ul> |

- ▷ *root\_squash* — Do not treat a remote root user as local root.
- ▷ *no\_root\_squash* — Treat a remote root user as local root. This is the default.
- ▶ **sync**: Values are: yes or no. When yes, the server cannot reply to requests until the changes made by the request are written to the disk. The default value is yes.
- ▶ **idmap**: Map remote id to local one, which can be used to improve the NFS security, for example, with *root\_squash*, remote root will be map to local anonuid/anongid.
- ▶ **secure**: Values are yes or no. If yes, all requests must originate from a port lower than 1024. If false, requests from any port number are accepted. The default is yes.
- ▶ **anonuid**: A numeric value that represents the anonymous UID. Sets the user ID of the anonymous account, to map all requests to one user.
- ▶ **anongid**: A numeric value that represents the anonymous GID. Sets the group ID of the anonymous account, to map all requests to one user.
- ▶ **squash\_uids**: A list of user IDs that are subject to anonymous mapping; for example: 0-15,20,25-50.
- ▶ **squash\_gids**: A list of group IDs that are subject to anonymous mapping; for example, 0-15,20,25-50.
- ▶ `sub_tenant_name=t1`

**Request Header**

```
POST /sub_tenant_admin/submit_add_node_nfs HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Cookie: _gui_session_id=48f4df73d20021e4716c4cd8fca9b0b7
Host: 10.5.116.244
Content-Length: 207
```

**Request Body**

```
node_uuid=564D85B4-8FA2-34BF-24A4-9BFB9C17A02B&subtenant_name=t1&share_path=efgh&host=test1&io=rw&squash=no_root_squash&sync=no&idmap=&secure=no&anonuid=&anongid=&squash_uids=&squash_gids=&sub_tenant_name=t1
```

**Response Header**

```
HTTP/1.1 200 OK
Connection: close
Date: Wed, 20 Jan 2010 11:50:31 GMT
Set-Cookie: _gui_session_id=48f4df73d20021e4716c4cd8fca9b0b7; path=/
Status: 200 OK
X-Runtime: 5062ms
ETag: "b98201b5ee2afcc512464a9c0842a7be"
Cache-Control: private, max-age=0, must-revalidate
Server: Mongrel 1.1.5
Content-Type: application/xml; charset=utf-8
Content-Length: 20
```

**Response Body**

```
<added>>true</added>
```

## Change NFS Shares

**Description** Lets you modify the configuration of an existing NFS share.

|                           |   |
|---------------------------|---|
| <b>Required Role</b>      | TenantAdmin   |
| <b>HTTP Method</b>        | POST  |
| <b>URI</b>                | /sub_tenant_admin/submit_configure_node_nfs   |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>HTTP Body:</p> <ul style="list-style-type: none"> <li>▶ <b>node_uuid:</b> The UUID of the NFS node to add.</li> <li>▶ <b>subtenant_name:</b></li> <li>▶ <b>share_path:</b> The root directory of the NFS access on the specified node. This is a directory to which the user of the service is to be given access. This should be the full path name of the directory from the root of the tenant namespace. <ul style="list-style-type: none"> <li><i>This path name must include the subtenant directory name, if you specified Multi Subtenant Access when you added this node to a tenant. For example, if the directory to be shared is <code>NFS_Share_A</code> and is within the namespace of a subtenant with ID <code>ad67a845aec4421badc97aabf66b4b08</code>, the share path should be entered as <code>ad67a845aec4421badc97aabf66b4b08/NFS_Share_A</code></i></li> </ul> </li> <li>▶ <b>host:</b> The hosts that access the share. Valid values are: <ul style="list-style-type: none"> <li>▷ <i>Single machine</i> — A fully qualified domain name (which can be resolved by the server), hostname (which can be resolved by the server), or IP address.</li> <li>▷ <i>Series of machines specified with wildcards</i> — Use the <code>*</code> or <code>?</code> character to specify a string match. Wildcards are not to be used with IP addresses; however, they may work accidentally if reverse DNS lookups fail. When specifying wildcards in fully qualified domain names, dots (<code>.</code>) are not included in the wildcard. For example, <code>*.example.com</code> includes <code>one.example.com</code> but does not include <code>one.two.example.com</code>.</li> <li>▷ <i>IP networks</i> — Use <code>a.b.c.d/z</code>, where <code>a.b.c.d</code> is the network and <code>z</code> is the number of bits in the netmask (for example, <code>192.168.0.0/24</code>). Another acceptable format is <code>a.b.c.d/netmask</code>, where <code>a.b.c.d</code> is the network and <code>netmask</code> is the netmask (for example, <code>192.168.100.8/255.255.0</code>).</li> <li>▷ <i>Netgroups</i> — Use the format <code>@group-name</code>, where <code>group-name</code> is the NIS netgroup name.</li> </ul> </li> <li>▶ <b>io:</b> Specify <code>r</code> or <code>rw</code>. Specifies whether the directory is read-only (<code>r</code>) or has read/write (<code>rw</code>) permissions. The default is <code>rw</code>.</li> <li>▶ <b>squash:</b> User access. Valid values are: <ul style="list-style-type: none"> <li>▷ <i>all_squash</i> — Treat all client users as anonymous users.</li> <li>▷ <i>root_squash</i> — Do not treat a remote root user as local root.</li> <li>▷ <i>no_root_squash</i> — Treat a remote root user as local root. This is the default.</li> </ul> </li> </ul> |

- ▶ **sync:** Values are: y (true) or n (false). When true, the server cannot reply to requests until the changes made by the request are written to the disk. The default value is y.
- ▶ **idmap:** Map remote id to local one, which can be used to improve the NFS security, for example, with root\_squash, remote root will be map to local anonuid/anongid.
- ▶ **secure:** Values are y(true) or n (false). If true, all requests must originate from a port lower than 1024. If false, requests from any port number are accepted. The default is y.
- ▶ **anonuid:** A numeric value that represents the anonymous UID. Sets the user ID of the anonymous account, to map all requests to one user.
- ▶ **anongid:** A numeric value that represents the anonymous GID. Sets the group ID of the anonymous account, to map all requests to one user.
- ▶ **squash\_uids:** A list of user IDs that are subject to anonymous mapping; for example: 0-15,20,25-50.
- ▶ **squash\_gids:** A list of group IDs that are subject to anonymous mapping; for example, 0-15,20,25-50.
- ▶ **sub\_tenant\_name:**

**Request Header**

```
POST /sub_tenant_admin/submit_configure_node_nfs HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Cookie: _gui_session_id=8cb62a5148719e998c2e28fa770c0c31
Host: 10.5.116.244
Content-Length: 204
```

**Request Body**

```
node_uuid=564D85B4-8FA2-34BF-24A4-9BFB9C17A02B&subtenant_name=t1&share_path=abcd&host=test1&io=rw&squash=root_squash&sync=no&idmap=&secure=no&anonuid=&anongid=&squash_uids=&squash_gids=&sub_tenant_name=t1
```

**Response Header**

```
HTTP/1.1 200 OK
Connection: close
Date: Wed, 20 Jan 2010 11:39:55 GMT
Set-Cookie: _gui_session_id=8cb62a5148719e998c2e28fa770c0c31; path=/
Status: 200 OK
X-Runtime: 6094ms
ETag: "27430fa0a0a46a8af142853e782c1a44"
Cache-Control: private, max-age=0, must-revalidate
Server: Mongrel 1.1.5
Content-Type: application/xml; charset=utf-8
Content-Length: 24
```

**Response Body**

```
<cleared>true</cleared>
```

## Delete NFS Shares

**Description** Lets you delete the specified NFS access path.

**Required Role** TenantAdmin

|                           |  |
|---------------------------|--|
| <b>HTTP Method</b>        | POST   |
| <b>URI</b>                | /sub_tenant_admin/submit_delete_node_nfs   |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>HTTP Body:</p> <ul style="list-style-type: none"> <li>▶ <b>node_uuid:</b> The UUID of the NFS node to add.</li> <li>▶ <b>subtenant_name:</b></li> <li>▶ <b>share_path:</b> The root directory of the NFS access on the specified node. This is a directory to which the user of the service is to be given access. This should be the full path name of the directory from the root of the tenant namespace.</li> </ul> <p><i>This path name must include the subtenant directory name, if you specified Multi Subtenant Access when you added this node to a tenant. For example, if the directory to be shared is NFS_Share_A and is within the namespace of a subtenant with ID ad67a845aec4421badc97aabf66b4b08, the share path should be entered as ad67a845aec4421badc97aabf66b4b08/NFS_Share_A</i></p> |
| <b>Request Header</b>     | <pre>POST /sub_tenant_admin/submit_delete_node_nfs HTTP/1.1 Accept: application/xml Content-Type: application/x-www-form-urlencoded Cookie: _gui_session_id=097049b68b4134933598423bd1a49900 Host: 10.5.116.244 Content-Length: 80</pre>   |
| <b>Request Body</b>       | <pre>node_uuid=564D85B4-8FA2-34BF-24A4-9BFB9C17A02B&amp;subtenant_name=t1&amp;share_path=abcd Description</pre>  |
| <b>Response Header</b>    | <pre>HTTP/1.1 200 OK Connection: close Date: Wed, 20 Jan 2010 09:01:10 GMT Set-Cookie: _gui_session_id=097049b68b4134933598423bd1a49900; path=/ Status: 200 OK X-Runtime: 2673ms ETag: "0d7da07534bfa39e841b8ee8cba2b4b2" Cache-Control: private, max-age=0, must-revalidate Server: Mongrel 1.1.5 Content-Type: application/xml; charset=utf-8 Content-Length: 24</pre>   |
| <b>Response Body</b>      | <deleted>>true</deleted>   |

## List NFS Shares

|                      |  |
|----------------------|--|
| <b>Description</b>   | Returns the list of NFS shares for the tenant. |
| <b>Required Role</b> | TenantAdmin                                    |

|                           |   |
|---------------------------|---|
| <b>HTTP Method</b>        | POST  |
| <b>URI</b>                | /sub_tenant_admin/node_nfs_list   |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>HTTP Body:</p> <ul style="list-style-type: none"> <li>▶ <b>node_uuid:</b> The UUID of the NFS node to add.</li> <li>▶ <b>subtenant_name:</b></li> <li>▶ <b>subtenant_name:</b></li> </ul>  |
| <b>Request Header</b>     | <pre>POST /sub_tenant_admin/node_nfs_list HTTP/1.1 Accept: application/xml Content-Type: application/x-www-form-urlencoded Cookie: _gui_session_id=54e0c8db2fe57fef33d069098915cfa6 Host: 10.5.116.244 Content-Length: 83</pre>   |
| <b>Request Body</b>       | node_uuid=564D85B4-8FA2-34BF-24A4-9BFB9C17A02B&subtenant_name=t1&sub_tenant_name=t1   |
| <b>Response Header</b>    | <pre>HTTP/1.1 200 OK Connection: close Date: Wed, 20 Jan 2010 12:37:27 GMT Set-Cookie: _gui_session_id=54e0c8db2fe57fef33d069098915cfa6; path=/ Status: 200 OK X-Runtime: 1805ms ETag: "d5286a8ca5df4cb02d35e9c4965ad5bf" Cache-Control: private, max-age=0, must-revalidate Server: Mongrel 1.1.5 Content-Type: application/xml; charset=utf-8 Content-Length: 401</pre>   |
| <b>Response Body</b>      | <pre>&lt;result&gt;   &lt;access_type&gt;nfs&lt;/access_type&gt;   &lt;nfs_node_list&gt;     &lt;nfs_node&gt;       &lt;share_path&gt;/mnt/mauifs/test&lt;/share_path&gt;       &lt;host&gt;test1&lt;/host&gt;     &lt;/nfs_node&gt;   &lt;nfs_node&gt;     &lt;share_path&gt;/mnt/mauifs/abcd&lt;/share_path&gt;     &lt;host&gt;host1&lt;/host&gt;   &lt;/nfs_node&gt;   &lt;nfs_node&gt;     &lt;share_path&gt;/mnt/mauifs/efgh&lt;/share_path&gt;     &lt;host&gt;test1&lt;/host&gt;   &lt;/nfs_node&gt; &lt;/nfs_node_list&gt; &lt;/result&gt;</pre> |

## Translate Node Name to UUID

|                           |  |
|---------------------------|--|
| <b>Description</b>        | Returns the translated UUID of the Node name you pass in.  |
| <b>Required Role</b>      | TenantAdmin  |
| <b>HTTP Method</b>        | GET  |
| <b>URI</b>                | /util/translate  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"><li>▶ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li></ul> <p>Querystring parameters:</p> <ul style="list-style-type: none"><li>▶ <b>scope:</b> Specify the type of conversion to perform on the specified value. For this request, the value is always <code>Node</code>.</li><li>▶ <b>input:</b> Specify the input type for the specified value. For this request, the value is always <code>name</code>.</li><li>▶ <b>value:</b> Specify the node name to convert.</li><li>▶ <b>output:</b> Specify the output type for the specified value. For this request, always specify <code>uuid</code>.</li></ul> |
| <b>Request Header</b>     | <pre>GET /util/translate?scope=Node&amp;input=name&amp;value=Testtest1-001&amp;output=uuid HTTP/1.1 Accept: application/xml Cookie: _gui_session_id=1c05b07ab4c4e08fa3d91f6b39035f86 Host: 10.5.116.110</pre>  |
| <b>Response Header</b>    | <pre>HTTP/1.1 200 OK Date: Fri, 24 Jul 2009 18:54:40 GMT Server: Mongrel 1.1.3 Status: 200 OK Cache-Control: no-cache Content-Type: application/xml; charset=utf-8 Content-Length: 68 Set-Cookie: _gui_session_id=1c05b07ab4c4e08fa3d91f6b39035f86; path=/ Connection: close</pre>   |
| <b>Response Body</b>      | <p>A successful request returns the node ID.</p> <pre>&lt;translated_id&gt;564DE0F3-A5BF-9DFD-8C5B-5EA182150521&lt;/translated_id&gt;</pre> <p>An unsuccessful request returns the following:</p> <pre>&lt;translated_id&gt;&lt;/translated_id&gt;</pre>   |

# 6

## Working with the Atmos Policy API

This section describes the policy selector and policy specification APIs. It includes these sections:

- ▶ [About Policy](#)
- ▶ [Working with Policies](#)
- ▶ [Policy Selectors API Reference](#)
- ▶ [Policy Specification API Reference](#)

---

### About Policy

*Policy management* involves defining a passive strategy for storing, retrieving, and managing objects and metadata. In the Atmos system, objects may include any type of binary content. Atmos relies on the Policy Manager to process this content, and it enables you to associate a policy with every object-management event. In addition, you associate metadata with objects as a way to understand their organization. As a result, you can achieve passive automation of object management for a variety of events, according to media type and related user or system metadata.

The TenantAdmin determines the nature of policies that run on the system. Policies implemented by a TenantAdmin apply only to the nodes defined for the tenant into which the TenantAdmin is placed. The TenantAdmin determines how object data is treated within Atmos based on a variety of conditions, including object types and events. Events associated with objects include creation, deletion, versioning, sorting, and updating. You can define discrete policies that automatically manage your stored data assets by specifying storage services, data transformations, placement strategies, workflows, and lifecycle management, all through the placement and definition of policies.

An Atmos policy is comprised of:

- ▶ A **policy specification** that defines the storage, retention, and deletion strategy for objects.
- ▶ A **policy selector** that defines the conditions that trigger when the policy is applied. When you create a policy selector, you bind it to an existing policy specification to create a complete policy.

Atmos policies have the following requirements and limitations:

- ▶ You can assign a maximum of 16 policies to one subtenant.
- ▶ Every policy must specify at least one replica, and a policy can specify an unlimited number of replicas. Any Atmos storage service can host a replica.
- ▶ Policies implemented by a TenantAdmin apply only to the nodes defined for the tenant where the TenantAdmin resides.

- ▶ Some objects may have one policy triggered on object (or metadata) creation and another triggered on object (or metadata) update. *If so, at least one synchronous replica in the create policy specification must exactly match a synchronous replica in the update policy specification, or the update fails.*

---

## Working with Policies

This section describes how to create a policy specification.

### Creating a Policy Specification

To create a policy, you need to decide:

- ▶ How to handle metadata
- ▶ How many replicas you want, and for each replica, you have to decide:
  - ▷ Is it sync or async.
  - ▷ Does it have a special characteristics such as striping, erasure coding, or federation.
  - ▷ What kind of read access should be used?
- ▶ Do you want to implement a retention or deletion rule?

The HTTP body of the create request includes parameters for values that define your response to each of the questions above. The following examples show how to supply these values. For a reference to all values, see [“Policy Specification API Reference” on page 92](#).

- ▶ [Example 1: Single Replica with Striping Enabled, Retention/Deletion Disabled](#)
- ▶ [Example 2: Single Replica with Erasure Coding Enabled](#)
- ▶ [Example 3 — Two replica with retention and deletion](#)

#### **Example 1: Single Replica with Striping Enabled, Retention/Deletion Disabled**

This example shows how to create a policy specification for a single replica with the following details:

| Specification  | Value/Description  | Parameter to use  |
|--|--|---|
| Policy Name  | Example1   | perform_action=create<br><br>other[specname]=Example1<br><br>entry[spec_name]=&<br><br>new_spec=true<br><br>replica_id= |
| Metadata   | sameAs, \$client   | metadata[1][location_modifier]=sameAs<br><br>metadata[1][location_place]=\$client                                       |
| Replica definition for Replica 1   | Replica Type: sync   | spec[1][type]=sync  |
|  | Enable Stripe: Yes   | spec[1][enable_stripe]=on   |
|  | Enable customize:no.   | spec[1][location_modifier]=sameAs   |
|  | But you must still supply the default values for replica location and Storage Server attribute parameters. | spec[1][location_place]=ANY<br>spec[1][ssattrs_placement]=OPTIMAL<br>spec[1][ssattrs_actions]=ANY                       |
|  | This is a replica with striping enabled so you must specify the following stripe attributes:               | spec[1][stripe_number]=4<br>spec[1][stripe_size]=40   |
|  | number, size, and units  | spec[1][stripe_units]=MB  |
| Federation and Erasure coding are not enabled for this replica, but you must still supply the parameters shown in the next column with the default values shown. | spec[1][service_name]=<br>spec[1][alg]=CRS<br>spec[1][fragmentation]=fragmentation1                        |   |
| Replica selection for read access  | geographic   | other[read_access]=geographic   |

| Specification  | Value/Description   | Parameter to use   |
|----------------|---|--|
| Retention      | Retention is not enabled for this replica, but you must still supply the retention parameters with no values. | other[retention_year]=<br>other[retention_month]=<br>other[retention_day]=<br>other[retention_hour]=<br>other[retention_minute]=<br>other[retention_second]= |
| Deletion       | Deletion is not enabled for this replica, but you must still supply the deletion parameters with no values.   | other[deletion_year]=<br>other[deletion_month]=<br>other[deletion_day]=<br>other[deletion_hour]=<br>other[deletion_minute]=<br>other[deletion_second]=       |
| Operation type |   | operation_type=sync  |

Once you have the specification described, you can build the HTTP request as described next:

- 1 Authenticate as a tenant administrator. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).
- 2 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 3 Use the HTTP POST verb with the following URI:

```
/tenant_admin/submit_create_spec
```

- 4 Build the HTTP body specifying the different policy specification attributes like this:

```
perform_action=create&other[specname]=Example1&entry[spec_name]=&new_spec=true&repli
ca_id=&metadata[1][location_modifier]=sameAs&metadata[1][location_place]=$client&spe
c[1][type]=sync&spec[1][enable_stripe]=on&spec[1][location_modifier]=sameAs&spec[1][
location_place]=ANY&spec[1][ssattrs_placement]=OPTIMAL&spec[1][ssattrs_actions]=ANY&
spec[1][stripe_number]=4&spec[1][stripe_size]=40&spec[1][stripe_units]=MB&spec[1][se
rvice_name]=&spec[1][alg]=CRS&spec[1][fragmentation]=fragmentation1&other[read_acces
s]=geographic&other[retention_year]=&other[retention_month]=&other[retention_day]=&o
ther[retention_hour]=&other[retention_minute]=&other[retention_second]=&other[deleti
on_year]=&other[deletion_month]=&other[deletion_day]=&other[deletion_hour]=&other[de
letion_minute]=&other[deletion_second]=&operation_type=sync
```

## Example 2: Single Replica with Erasure Coding Enabled

The following example shows how to build a policy for a single sync replica with erasure coding enabled. The erasure coding algorithm is CRS (Cauchy Reed-Solomon), and the data fragmentation is 9:3 (data=9, code=3).

| Specification   | Value/Description  | Parameter to use  |                                    |
|---|--|---|------------------------------------|
| Policy Name   | ExampleErasure   | perform_action=create<br><br>other[specname]=ExampleErasure<br><br>entry[spec_name]=&<br><br>new_spec=true<br><br>replica_id= |                                    |
| Metadata  | sameAs, \$client   | metadata[1][location_modifier]=sameAs<br><br>metadata[1][location_place]=\$client   |                                    |
| Replica definition for Replica 1  | Replica Type: sync   | spec[1][type]=sync  |                                    |
|   | Erasure coding=yes   | spec[1][enable_erasure]=on  |                                    |
|   | Enable customize:no.   | spec[1][location_modifier]=sameAs   |                                    |
|   | But you must still supply the default values for replica location and Storage Server attribute parameters.   |   | spec[1][location_place]=ANY        |
|   |  |   | spec[1][ssattrs_placement]=OPTIMAL |
|   |  |   | spec[1][ssattrs_actions]=ANY       |
|   | This is a replica without striping enabled, but you must still supply the striping attributes with no values |   | spec[1][stripe_number]=            |
|   |  | spec[1][stripe_size]=   |                                    |
|   |  | spec[1][stripe_units]=B   |                                    |
| Federation is not enabled; however, you must supply the service_name with not value.  |  | spec[1][service_name]=  |                                    |
| Erasure coding is enabled. The algorithm value is always CRS. The fragmentation value can be fragmentation1 to represent 9:3. |  | spec[1][alg]=CRS  |                                    |
|   |  | spec[1][fragmentation]=fragmentation1   |                                    |
| Replica selection for read access   | geographic   | other[read_access]=geographic   |                                    |

| Specification  | Value/Description   | Parameter to use   |
|----------------|---|--|
| Retention      | Retention is not enabled for this replica, but you must still supply the retention parameters with no values. | other[retention_year]=<br>other[retention_month]=<br>other[retention_day]=<br>other[retention_hour]=<br>other[retention_minute]=<br>other[retention_second]= |
| Deletion       | Deletion is not enabled for this replica, but you must still supply the deletion parameters with no values.   | other[deletion_year]=<br>other[deletion_month]=<br>other[deletion_day]=<br>other[deletion_hour]=<br>other[deletion_minute]=<br>other[deletion_second]=       |
| Operation type |   | operation_type=sync  |

Once you have the specification described, you can build the HTTP request as described next:

- 1 Authenticate as a tenant administrator. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).
- 2 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 3 Use the HTTP POST verb with the following URI:

```
/tenant_admin/submit_create_spec
```

- 4 Build the HTTP body specifying the different policy specification attributes like this:

```
perform_action=create&other[specname]=ExampleErasure&entry[spec_name]=&&new_spec=true&replica_id=&metadata[1][location_modifier]=sameAs&metadata[1][location_place]=$client&spec[1][type]=sync&spec[1][enable_erasure]=on&spec[1][location_modifier]=sameAs&spec[1][location_place]=ANY&spec[1][ssattrs_placement]=OPTIMAL&spec[1][ssattrs_actions]=ANY&spec[1][stripe_number]=&spec[1][stripe_size]=&spec[1][stripe_units]=B&spec[1][service_name]=&spec[1][alg]=CRS&spec[1][fragmentation]=fragmentation1&other[read_access]=geographic&other[retention_year]=&other[retention_month]=&other[retention_day]=&other[retention_hour]=&other[retention_minute]=&other[retention_second]=&other[deletion_year]=&other[deletion_month]=&other[deletion_day]=&other[deletion_hour]=&other[deletion_minute]=&other[deletion_second]=&operation_type=sync
```

### Example 3 — Two replica with retention and deletion

The following example shows how to build a policy that defines 2 replicas (1 async, 1 sync). Retention is enabled, but it has no other special properties associated with it.

| Specification | Value/Description  | Parameter to use  |
|---------------|--|---|
| Policy Name   | ExampleCustom2   | perform_action=create<br>other[specname]=ExampleCustom2<br>entry[spec_name]=&<br>new_spec=true<br>replica_id=   |
| Metadata      | sameAs, \$client   | metadata[1][location_modifier]=sameAs<br>metadata[1][location_place]=\$client   |
| Replica 1     | Replica Type: async<br>Enable customize: no<br>But you must still supply the default values for replica location and Storage Server attribute parameters.<br>This is a replica without striping enabled, but you must still supply the striping attributes with no values<br>Federation is not enabled; however, you must supply the service_name with not value.<br>Erasure coding is not enabled., but you must still supply the alg and fragmentation parameters with defaults. | spec[1][type]=async<br>spec[1][location_modifier]=sameAs<br>spec[1][location_place]=ANY<br>spec[1][ssattrs_placement]=OPTIMAL<br>spec[1][ssattrs_actions]=ANY<br>spec[1][stripe_number]=<br>spec[1][stripe_size]=<br>spec[1][stripe_units]=B<br>spec[1][service_name]=<br>spec[1][alg]=CRS<br>spec[1][fragmentation]=fragmentation1 |

| <b>Specification</b>  | <b>Value/Description</b>   | <b>Parameter to use</b>            |
|---|--|------------------------------------|
| Replica 2   | Replica Type: async  | spec[2][type]=sync                 |
|   | Enable customize: no   | spec[2][location_modifier]=sameAs  |
|   | But you must still supply the default values for replica location and Storage Server attribute parameters.   | spec[2][location_place]=ANY        |
|   |  | spec[2][ssattrs_placement]=OPTIMAL |
|   |  | spec[2][ssattrs_actions]=ANY       |
|   | This is a replica without striping enabled, but you must still supply the striping attributes with no values | spec[2][stripe_number]=            |
| spec[2][stripe_size]=   |  |                                    |
| spec[2][stripe_units]=B   |  |                                    |
| Federation is not enabled; however, you must supply the service_name with not value.                          | spec[2][service_name]=   |                                    |
| Erasure coding is not enabled., but you must still supply the alg and fragmentation parameters with defaults. | spec[2][alg]=CRS   |                                    |
|   | spec[2][fragmentation]=fragmentation1  |                                    |
| Replica selection for read access   | geographic   | other[read_access]=geographic      |
| Retention   | Retention is enabled.  | other[enable_retention]=on         |
|   |  | other[retention_year]=             |
|   |  | other[retention_month]=            |
|   |  | other[retention_day]=5             |
|   |  | other[retention_hour]=             |
|   |  | other[retention_minute]=           |
|   |  | other[retention_second]=35         |

| Specification  | Value/Description    | Parameter to use   |
|----------------|----------------------|--|
| Deletion       | Deletion is enabled. | other[enable_deletion]=on<br><br>other[deletion_year]=<br><br>other[deletion_month]=6<br><br>other[deletion_day]=<br><br>other[deletion_hour]=<br><br>other[deletion_minute]=<br><br>other[deletion_second]= |
| Operation type |                      | operation_type=sync  |

Once you have the specification described, you can build the HTTP request as described next:

- 1 Authenticate as a tenant administrator. For more information, see [“Authenticate as a TenantAdmin” on page 10](#).
- 2 Obtain and use the session ID for all subsequent calls by specifying it in the HTTP Header Cookie parameter. You set **Cookie** to the `_gui_session_id` returned by the authentication method.
- 3 Use the HTTP POST verb with the following URI:

```
/tenant_admin/submit_create_spec
```

- 4 Build the HTTP body specifying the different policy specification attributes like this:

```
perform_action=create&other[specname]=ExampleCustom2&entry[spec_name]=&&new_spec=true&replica_id=&metadata[1][location_modifier]=sameAs&metadata[1][location_place]=$client&spec[1][type]=async&spec[1][location_modifier]=sameAs&spec[1][location_place]=ANY&spec[1][ssattrs_placement]=OPTIMAL&spec[1][ssattrs_actions]=ANY&spec[1][stripe_number]=&spec[1][stripe_size]=&spec[1][stripe_units]=B&spec[1][service_name]=&spec[1][alg]=CRS&spec[1][fragmentation]=fragmentation1&spec[2][type]=sync&spec[2][location_modifier]=sameAs&spec[2][location_place]=ANY&spec[2][ssattrs_placement]=OPTIMAL&spec[2][ssattrs_actions]=ANY&spec[2][stripe_number]=&spec[2][stripe_size]=&spec[2][stripe_units]=B&spec[2][service_name]=&spec[2][alg]=CRS&spec[2][fragmentation]=fragmentation1&other[read_access]=geographic&other[enable_retention]=on&other[retention_year]=&other[retention_month]=&other[retention_day]=5&other[retention_hour]=&other[retention_minute]=&other[retention_second]=35&other[enable_deletion]=on&other[deletion_year]=&other[deletion_month]=6&other[deletion_day]=&other[deletion_hour]=&other[deletion_minute]=&other[deletion_second]=&operation_type=sync
```

## Policy Selectors API Reference

The Policy Selector API includes:

- ▶ [Assign a Policy Selector](#)

- ▶ [Create a Policy Selector](#)
- ▶ [Delete a Policy Selector](#)
- ▶ [Get a Policy Selector](#)
- ▶ [List Policy Selectors](#)
- ▶ [Modify a Policy Selector](#)
- ▶ [Remove a Policy Selector](#)

## Assign a Policy Selector

|                           |  |
|---------------------------|--|
| <b>Description</b>        | Assigns the specified policy selector to the specified subtenant(s).The HTTP request header must include the <code>Cookie</code> with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication. The request body must specify the policy selector name and the name of the subtenant to assign the policy to.   |
| <b>Required Role</b>      | TenantAdmin  |
| <b>HTTP Method</b>        | POST   |
| <b>URI</b>                | <code>/tenant_admin/submit_assign_policy</code>  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▶ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>HTTP body:</p> <ul style="list-style-type: none"> <li>▶ <b>selector:</b> Specify the policy selector name.</li> <li>▶ <b>policy_type:</b> Optional.</li> <li>▶ <b>operation_type:</b> create <ul style="list-style-type: none"> <li>▷ <code>sync:</code> default.</li> <li>▷ <code>async:</code></li> </ul> </li> <li>▶ <b>subtenants:</b> Specify the name of the subtenant to assign the policy selector to.</li> </ul> |
| <b>Request Header</b>     | <pre>POST /tenant_admin/submit_assign_policy HTTP/1.1 Accept: application/xml Content-Type: application/x-www-form-urlencoded Cookie: _gui_session_id=4bb29923e416a851e2204f3fccc3f73d Host: 10.5.116.244 Content-Length: 64</pre>   |

**Request Body** selector=policyselector2&policy\_type=0&operation\_type=async&subtenants[]=t1  
or  
selector=policyselector2&policy\_type=0&operation\_type=sync&subtenants[]=t1

**Response Header** HTTP/1.1 200 OK  
Date: Wed, 16 Sep 2009 05:21:39 GMT  
Server: Mongrel 1.1.3  
Status: 200 OK  
Cache-Control: no-cache  
Content-Type: application/xml; charset=utf-8  
Content-Length: 24  
Set-Cookie: \_gui\_session\_id=4bb29923e416a851e2204f3fccc3f73d; path=/  
Connection: close

**Response Body** A successful request returns:  
  
<cleared>true</cleared>

## Create a Policy Selector

**Description** Creates a policy selector and binds it to an existing policy specification. You can define a policy selector to act on user metadata, system metadata, or an XQuery specification.

The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role** TenantAdmin

**HTTP Method** POST

**URI** /tenant\_admin/submit\_create\_selector

**Request Parameters** HTTP header:  
  
▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

HTTP body:

| parameter      | Description  |
|----------------|--|
| perform_action | Specify <code>create</code> (for create operations) or <code>modify</code> (for modify operations).<br><br><b>Example:</b> perform_action=create |
| entry_index    | Not used. Supply the parameter with no value.<br><br><b>Example:</b> entry_index=  |

| <b>parameter</b>       | <b>Description</b>   |
|------------------------|--|
| entry[edit_mode]       | Specify what type of data the policy selector you are creating. Valid values: 0 (user metadata), 1 (system metadata), 2 (advanced).<br><br><b>Example:</b> entry[edit_mode]=1  |
| entry[spec_name]       | Not used. Supply the parameter with no value.<br><br><b>Example:</b> entry[spec_name]=   |
| entry[event]           | Not used. Supply the parameter with no value.<br><br><b>Example:</b> entry[event]=   |
| entry[id]              | Specify the name of the selector to create or modify.<br><br><b>Example:</b> entry[id]=ps2   |
| entry[specname]        | Specify the name of an existing policy specification. This binds the policy selector to the policy specification.<br><br><b>Example:</b> other[specname]=default   |
| entry[user_field_name] | Specify a user-metadata tag name. Use with user_operator and user_value. The value of edit_mode must be 0.<br><br><b>Example:</b> entry[user_field_name]=  |
| entry[user_operator]   | Specify a value comparison operator. Valid values: EQUALS, ENDS+WITH, STARTS+WITH, CONTAINS, and the following symbols: <, =, >, <=, >=<br><br>Use with user_operator and user_value. The value of edit_mode must be 0.<br><br><b>Example:</b> entry[user_operator]=EQUALS |
| entry[user_value]      | Specify a user metadata value. The value can be text, a number or a date that matches the value specified in the user_field_name parameter.<br><br>Use with user_operator and user_field_name. The value of edit_mode must be 0.<br><br><b>Example:</b> entry[user_value]= |
| entry[user_event]      | Specify a user metadata trigger event to activate this policy selector. Valid values: ON_CREATE, ON_UMD_UPDATE, ON_SMD_UPDATE. The value of edit_mode must be 0.<br><br><b>Example:</b> entry[user_event]=ON_CREATE  |

| parameter             | Description   |
|-----------------------|---|
| entry[sys_field_name] | <p>Specify a system metadata tag name. Valid values are: atime, mtime, ctime, itime, uid, gid, size, objname. Use with sys_operator, sys_value, and sys_event. The value of edit_mode must be 1.</p> <p><b>Example:</b> entry[sys_field_name]=objname</p>   |
| entry[sys_operator]   | <p>Specify a system-metadata-value comparison operator. Valid values: EQUALS, ENDS+WITH, STARTS+WITH, CONTAINS, and the symbols &lt;,=,&gt;, &lt;=,&gt;=</p> <p>Use with sys_field_name, sys_value, and sys_event. The value of edit_mode must be 1.</p> <p><b>Example:</b> entry[sys_operator]=ENDS+WITH</p>     |
| entry[sys_value]      | <p>Specify a system metadata value. The value can be text, a number or a date that matches the value specified in the sys_field_name parameter.</p> <p>Use with sys_field_name, sys_operator, and sys_event. The value of edit_mode must be 1.</p> <p><b>Example:</b> entry[sys_value]=basic_2_replica_2s_sub</p> |
| entry[sys_event]      | <p>Specify a system metadata trigger event for policy activation. Valid values: ON_CREATE, ON_UMD_UPDATE, ON_SMD_UPDATE).</p> <p>Use with sys_field_name, sys_operator, and sys_value. The value of edit_mode must be 1.</p> <p><b>Example:</b> entry[sys_event]=ON_CREATE</p>                                    |
| entry[xquery]         | <p>Specify an Xquery string against which to match objects.</p> <p>Use with adv_event. The value of edit_mode must be 2.</p> <p><b>Example:</b><br/>entry[xquery]='for+%24c+in+collection%28%29+return+%24c'</p> <p>The example represents the escaped string for 'for \$c in collection() return \$c'</p>        |
| entry[adv_event]      | <p>Specify an Xquery trigger event for policy activation. Valid values: ON_CREATE (non-system-metadata modes), ON_UMD_UPDATE, ON_SMD_UPDATE.</p> <p>Use with xquery. The value of edit_mode must be 2.</p> <p><b>Example:</b> entry[adv_event]=ON_CREATE</p>  |
| operation_type        | <p>Values are: sync (the default) or async.</p>   |

This example shows how to create a policy selector named `ps2` for the default policy. It operates on system metadata `ON_CREATE` event. When the objname ends with the extension `.basic_2_replica_2s_sub`.

**Request Header**

```
POST /tenant_admin/submit_create_selector HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Cookie: _gui_session_id=f01b73beb4d2bc0b4171d76da7e260e9
Host: 10.5.116.244
Content-Length: 446
```

**Request Body**

```
perform_action=create&entry_index=&entry[edit_mode]=1&entry[spec_name]=&entry[event]=&entry[id]=ps2&other[specname]=default&entry[user_field_name]=&entry[user_operator]=EQUALS&entry[user_value]=&entry[user_event]=ON_CREATE&entry[sys_field_name]=objname&entry[sys_operator]=ENDS+WITH&entry[sys_value]=.basic_2_replica_2s_sub&entry[sys_event]=ON_CREATE&entry[xquery]=&entry[adv_event]=ON_CREATE&operation_type=async
```

**Response Header**

```
HTTP/1.1 200 OK
Date: Wed, 16 Sep 2009 06:47:59 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 20
Set-Cookie: _gui_session_id=f01b73beb4d2bc0b4171d76da7e260e9; path=/
Connection: close
```

**Response Body**

A successful request returns:

```
<added>true</added>
```

## Delete a Policy Selector

**Description**

Deletes the specified policy selector. The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role**

TenantAdmin

**HTTP Method**

GET

**URI**

`/tenant_admin/delete_policy_selector`

**Request Parameters**

HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

QueryString parameters:

- ▷ **selector:** The name of the policy selector to delete.
- ▷ **operation\_type:** Optional. Value can be sync or async. Default is sync.

**Request Header** GET /tenant\_admin/delete\_policy\_selector?selector=SampleSelector&operation\_type=sync  
HTTP/1.1  
Accept: application/xml  
Cookie: \_gui\_session\_id=7cd0d3db7997818c906092ff4fad1a72  
Host: 10.5.116.244

or

GET  
/tenant\_admin/delete\_policy\_selector?selector=policyselector2&operation\_type=async  
HTTP/1.1  
Accept: application/xml  
Cookie: \_gui\_session\_id=43c03c060654199e52f101a5df0bab1d  
Host: 10.5.116.244

**Request Body** None

**Response Header** HTTP/1.1 200 OK  
Date: Wed, 16 Sep 2009 06:40:35 GMT  
Server: Mongrel 1.1.3  
Status: 200 OK  
Cache-Control: no-cache  
Content-Type: application/xml; charset=utf-8  
Content-Length: 24  
Set-Cookie: \_gui\_session\_id=7cd0d3db7997818c906092ff4fad1a72; path=/  
Connection: close

**Response Body** A successful request returns:

<deleted>true</deleted>

## Get a Policy Selector

**Description** Retrieves details about the specified policy. The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

Details include:

- ▶ Policy name
- ▶ Status
- ▶ Edit mode
- ▶ Metadata tag
- ▶ Match operator
- ▶ Metadata value
- ▶ On event

**Required Role** TenantAdmin

|                           |  |
|---------------------------|--|
| <b>HTTP Method</b>        | GET  |
| <b>URI</b>                | /tenant_admin/get_policy_selector  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>Querystring parameters</p> <ul style="list-style-type: none"> <li>▷ <b>selector:</b> Specify the selector name.</li> </ul>  |
| <b>Request Header</b>     | <pre>GET /tenant_admin/get_policy_selector?selector=policyselector1 HTTP/1.1 Accept: application/xml Cookie: _gui_session_id=468042d911f7268cebda909bfa2ba10e Host: 10.5.116.244</pre>   |
| <b>Request Body</b>       | None   |
| <b>Response Header</b>    | <pre>HTTP/1.1 200 OK Date: Wed, 16 Sep 2009 06:10:19 GMT Server: Mongrel 1.1.3 Status: 200 OK Cache-Control: no-cache Content-Type: application/xml; charset=utf-8 Content-Length: 282 Set-Cookie: _gui_session_id=468042d911f7268cebda909bfa2ba10e; path=/ Connection: close</pre>  |
| <b>Response Body</b>      | <p>A successful request returns:</p> <pre>&lt;policy_selector&gt;   &lt;name&gt;policyselector1&lt;/name&gt;   &lt;status&gt;Completed&lt;/status&gt;   &lt;policy_id&gt;&lt;/policy_id&gt;   &lt;edit_mode&gt;systemmd&lt;/edit_mode&gt;   &lt;metadata_tag&gt;atime&lt;/metadata_tag&gt;   &lt;match_operator&gt;&amp;gt;=&lt;/match_operator&gt;   &lt;metadata_value&gt;2010-01-08 00:00:00&lt;/metadata_value&gt;   &lt;on_event&gt;ON_CREATE&lt;/on_event&gt; &lt;/policy_selector&gt;</pre> |

## List Policy Selectors

|                      |  |
|----------------------|--|
| <b>Description</b>   | Retrieves general information about the policy selectors associated with a tenant or subtenant. The HTTP request header must include the <code>Cookie</code> with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication. |
| <b>Required Role</b> | TenantAdmin  |
| <b>HTTP Method</b>   | GET  |

**URI** /tenant\_admin/list\_policy\_selector

**Request Parameters** HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

Querystring parameters:

- ▷ **sub\_tenant\_name:** (Optional.) specify the name of the subtenant if you want to narrow the result to only the policy selectors assigned to the subtenant. For example: `GET /tenant_admin/list_policy_selector?sub_tenant_name=t1`

**Request Header** GET /tenant\_admin/list\_policy\_selector HTTP/1.1  
Accept: application/xml  
Cookie: \_gui\_session\_id=0537b8f8f361c03c47cc6975d04d4a9c  
Host: 10.5.116.244

**Request Body** None

**Response Header** HTTP/1.1 200 OK  
Date: Wed, 16 Sep 2009 06:16:11 GMT  
Server: Mongrel 1.1.3  
Status: 200 OK  
Cache-Control: no-cache  
Content-Type: application/xml; charset=utf-8  
Content-Length: 992  
Set-Cookie: \_gui\_session\_id=0537b8f8f361c03c47cc6975d04d4a9c; path=/  
Connection: close

**Response Body** A successful request returns:

```
<policy_selector_list>
  <policy_selector>
    <name>Cust1SizeSelector</name>
    <expression>size > 5000000</expression>
    <on_event>ON_CREATE</on_event>
    <spec>1LSand1RA</spec>
  </policy_selector>
  <policy_selector>
    <name>Cust1UIDSelector</name>
    <expression>uid equals Customer1UID</expression>
    <on_event>ON_CREATE</on_event>
    <spec>2LSand1RA</spec>
  </policy_selector>
  <policy_selector>
    <name>Cust2Selector</name>
    <expression>uid equals Customer2UID</expression>
    <on_event>ON_CREATE</on_event>
  </policy_selector>
</policy_selector_list>
```

```

        <spec>1LSand1RA</spec>
    </policy_selector>
    <policy_selector>
        <name>DevSelector</name>
        <expression>uid equals DevUID</expression>
        <on_event>ON_CREATE</on_event>
        <spec>1LSand1LScomp</spec>
    </policy_selector>
    <policy_selector>
        <name>SampleSelector</name>
        <expression>atime equals 2009-09-16 00:00:00</expression>
        <on_event>ON_CREATE</on_event>
        <spec>SamplePolicy</spec>
    </policy_selector>
</policy_selector_list>

```

## Modify a Policy Selector

**Description** Modifies the specified policy selector. A modify operation replaces the existing policy selector. The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role** TenantAdmin

**HTTP Method** POST

**URI** `/tenant_admin/submit_modify_selector`

**Request Parameters** HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

HTTP body:

| parameter        | Description  |
|------------------|--|
| perform_action   | Specify <code>create</code> (for create operations) or <code>modify</code> (for modify operations).<br><br><b>Example:</b> <code>perform_action=modify</code>                              |
| entry_index      | Not used. Supply the parameter with no value.<br><br><b>Example:</b> <code>entry_index=</code>   |
| entry[edit_mode] | Specify what type of data the policy selector you are creating. Valid values: 0 (user metadata), 1 (system metadata), 2 (advanced).<br><br><b>Example:</b> <code>entry[edit_mode]=1</code> |

| <b>parameter</b>       | <b>Description</b>   |
|------------------------|--|
| entry[spec_name]       | Not used. Supply the parameter with no value.<br><br><b>Example:</b> entry[spec_name]=   |
| entry[event]           | Not used. Supply the parameter with no value.<br><br><b>Example:</b> entry[event]=   |
| entry[id]              | Specify the name of the selector to create or modify.<br><br><b>Example:</b> entry[id]=ps2   |
| entry[specname]        | Specify the name of an existing policy specification. This binds the policy selector to the policy specification.<br><br><b>Example:</b> other[specname]=default   |
| entry[user_field_name] | Specify a user-metadata tag name. Use with user_operator and user_value. The value of edit_mode must be 0.<br><br><b>Example:</b> entry[user_field_name]=  |
| entry[user_operator]   | Specify a value comparison operator. Valid values: EQUALS, ENDS+WITH, STARTS+WITH, CONTAINS, and the following symbols: <, =, >, <=, >=<br><br>Use with user_operator and user_value. The value of edit_mode must be 0.<br><br><b>Example:</b> entry[user_operator]=EQUALS |
| entry[user_value]      | Specify a user metadata value. The value can be text, a number or a date that matches the value specified in the user_field_name parameter.<br><br>Use with user_operator and user_field_name. The value of edit_mode must be 0.<br><br><b>Example:</b> entry[user_value]= |
| entry[user_event]      | Specify a user metadata trigger event to activate this policy selector. Valid values: ON_CREATE, ON_UMD_UPDATE, ON_SMD_UPDATE. The value of edit_mode must be 0.<br><br><b>Example:</b> entry[user_event]=ON_CREATE  |
| entry[sys_field_name]  | Specify a system metadata tag name. Valid values are: atime, mtime, ctime, itime, uid, gid, size, objname. Use with sys_operator, sys_value, and sys_event. The value of edit_mode must be 1.<br><br><b>Example:</b> entry[sys_field_name]=objname                         |

| parameter           | Description   |
|---------------------|---|
| entry[sys_operator] | <p>Specify a system-metadata-value comparison operator. Valid values: EQUALS, ENDS+WITH, STARTS+WITH, CONTAINS, and the symbols &lt;=, &gt;, &lt;=, &gt;=</p> <p>Use with <code>sys_field_name</code>, <code>sys_value</code>, and <code>sys_event</code>. The value of <code>edit_mode</code> must be 1.</p> <p><b>Example:</b> <code>entry[sys_operator]=ENDS+WITH</code></p>                 |
| entry[sys_value]    | <p>Specify a system metadata value. The value can be text, a number or a date that matches the value specified in the <code>sys_field_name</code> parameter.</p> <p>Use with <code>sys_field_name</code>, <code>sys_operator</code>, and <code>sys_event</code>. The value of <code>edit_mode</code> must be 1.</p> <p><b>Example:</b> <code>entry[sys_value]=basic_2_replica_2s_sub</code></p> |
| entry[sys_event]    | <p>Specify a system metadata trigger event for policy activation. Valid values: ON_CREATE, ON_UMD_UPDATE, ON_SMD_UPDATE).</p> <p>Use with <code>sys_field_name</code>, <code>sys_operator</code>, and <code>sys_value</code>. The value of <code>edit_mode</code> must be 1.</p> <p><b>Example:</b> <code>entry[sys_event]=ON_CREATE</code></p>   |
| entry[xquery]       | <p>Specify an Xquery string against which to match objects. Use with <code>adv_event</code>. The value of <code>edit_mode</code> must be 2.</p> <p><b>Example:</b><br/> <code>entry[xquery]='for+%24c+in+collection%28%29+return+%24c'</code></p> <p>The example represents the escaped string for 'for \$c in collection() return \$c'</p>   |
| entry[adv_event]    | <p>Specify an Xquery trigger event for policy activation. Valid values: ON_CREATE (non-system-metadata modes), ON_UMD_UPDATE, ON_SMD_UPDATE.</p> <p>Use with <code>xquery</code>. The value of <code>edit_mode</code> must be 2.</p> <p><b>Example:</b> <code>entry[adv_event]=ON_CREATE</code></p>   |
| operation_type      | <p>Values are:</p> <ul style="list-style-type: none"> <li>▶ <b>sync:</b> the default.</li> <li>▶ <b>async:</b></li> </ul>   |

This example shows how to modify a policy selector named `ps2` for the default policy. It operates on system metadata `ON_CREATE` event. When the objname ends with the extension `.basic_2_replica_2s_sub`.

|                        |   |
|------------------------|---|
| <b>Request Header</b>  | <pre>POST /tenant_admin/submit_modify_selector HTTP/1.1 Accept: application/xml Content-Type: application/x-www-form-urlencoded Cookie: _gui_session_id=d273fc48ccea1ac20a7fdcdf69434267 Host: 10.5.116.244 Content-Length: 451</pre>   |
| <b>Request Body</b>    | <pre>perform_action=modify&amp;entry_index=&amp;entry[edit_mode]=1&amp;entry[spec_name]=&amp;entry[event]=&amp;entry[id]=ps2&amp;other[specname]=default&amp;entry[user_field_name]=&amp;entry[user_operator]=EQUALS&amp;entry[user_value]=&amp;entry[user_event]=ON_CREATE&amp;entry[sys_field_name]=objname&amp;entry[sys_operator]=ENDS+WITH&amp;entry[sys_value]=.basic_2_replica_2s_sub&amp;entry[sys_event]=ON_CREATE&amp;entry[xquery]=&amp;entry[adv_event]=ON_CREATE</pre> |
| <b>Response Header</b> | <pre>HTTP/1.1 200 OK Date: Wed, 16 Sep 2009 06:52:58 GMT Server: Mongrel 1.1.3 Status: 200 OK Cache-Control: no-cache Content-Type: application/xml; charset=utf-8 Content-Length: 24 Set-Cookie: _gui_session_id=d273fc48ccea1ac20a7fdcdf69434267; path=/ Connection: close</pre>  |
| <b>Response Body</b>   | <p>A successful request returns:</p> <pre>&lt;cleared&gt;true&lt;/cleared&gt;</pre>   |

## Remove a Policy Selector

|                           |   |
|---------------------------|---|
| <b>Description</b>        | Removes the specified policy selector from all subtenants. The HTTP request header must include the Cookie with the session ID ( <code>_gui_session_id</code> ) returned from a successful authentication.  |
| <b>Required Role</b>      | TenantAdmin   |
| <b>HTTP Method</b>        | POST  |
| <b>URI</b>                | <code>/tenant_admin/submit_assign_policy</code>   |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>HTTP body:</p> <ul style="list-style-type: none"> <li>▷ <b>selector:</b> Specify the policy selector name.</li> <li>▷ <b>policy_type:</b> Optional.</li> </ul> |
| <b>Request Header</b>     | <pre>POST /tenant_admin/submit_assign_policy HTTP/1.1 Accept: application/xml Content-Type: application/x-www-form-urlencoded Cookie: _gui_session_id=4fe718ff82a9c1166c3386a90ebd311f Host: 10.5.116.244 Content-Length: 37</pre>  |

|                        |  |
|------------------------|--|
| <b>Request Body</b>    | <code>selector=SampleSelector&amp;policy_type=0</code>   |
| <b>Response Header</b> | <pre> HTTP/1.1 200 OK Date: Wed, 16 Sep 2009 06:25:05 GMT Server: Mongrel 1.1.3 Status: 200 OK Cache-Control: no-cache Content-Type: application/xml; charset=utf-8 Content-Length: 24 Set-Cookie: _gui_session_id=4fe718ff82a9c1166c3386a90ebd311f; path=/ Connection: close </pre> |
| <b>Response Body</b>   | <p>A successful request returns:</p> <pre>&lt;cleared&gt;true&lt;/cleared&gt;</pre>  |

---

## Policy Specification API Reference

The policy specification The Policy Specification API includes:

- ▶ [Create a Policy Specification](#)
- ▶ [Delete a Policy Specification](#)
- ▶ [Get a Policy Specification](#)
- ▶ [List Policy Specifications](#)
- ▶ [Modify a Policy Specification](#)

### Create a Policy Specification

**Description** Creates a new policy. Returns true if the request is successful; otherwise returns an error message.

The HTTP body for the policy specification is made up of the following parts:

- ▶ **General information** which define such things as the operation to perform (modify or create) and the policy specification name.
- ▶ **Metadata replica descriptor** that defines how to replicate the metadata such as location modifier and the location place.
- ▶ **Replica descriptors.** Each replica descriptor includes the attribute (synchronous or asynchronous), location constraints, and requested storage-service capabilities. The location-constraints descriptor includes a place label and location modifier, which indicates the requested relation with the specified location; for example, the same as the specified location label or different than the given location
- ▶ **Define user data replicas.** Because you can define any number of data replicas, this is an array, and every element the array represents a single replica. The data replication part has several subparts

where you define how data is written (sync or async), whether to use striping, federation, or erasure coding.

- **Read, retention, and deletion rules** define how to read, retain, and delete the data.

The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role**

TenantAdmin

**HTTP Method**

POST

**URI**

POST /tenant\_admin/submit\_create\_spec

**Request Parameters**

HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

HTTP body:

The following table describes the **general parameters**.

| Parameter        | Description  |
|------------------|--|
| perform_action   | Specify <code>create</code> (for create operations) or <code>modify</code> (for modify operations).<br><br><b>Example:</b> <code>perform_action=create</code>                              |
| other[specname]  | Specify the name of the policy to create or modify. The name can be any string, but it cannot include special characters.<br><br><b>Example:</b> <code>other[specname]=samplepolicy</code> |
| entry[spec_name] | Not used. Supply the parameter with no value.  |
| new_spec         | Specify <code>true</code> for all create and modify operations.<br><br><b>Example:</b> <code>new_spec=true</code>  |
| replica_id       | Not used. Supply the parameter with no value.  |

Use the parameters in the following table to define the metadata storage.

| Parameter                      | Description  |
|--------------------------------|--|
| metadata[x][location_modifier] | <p>Specifies the matching criteria for the location_place. [x] indicates replica number. For metadata the value is always 1.</p> <p>Valid values are: <code>sameAs</code> (no replica customization), and <code>otherThan</code>.</p> <p>You cannot combine the location_modifier <code>otherThan</code> with the location_place of <code>ANY</code>.</p> <p><b>Example:</b> <code>metadata[1][location_modifier]=sameAs</code></p>  |
| metadata[x][location_place]    | <p>Specifies the location where Atmos should create new files and directories. [x] indicates replica number. For metadata the value is always 1.</p> <p>Valid values:</p> <ul style="list-style-type: none"> <li>▶ <b>ANY</b> (the default location): For the Web-service object interface, this is a location at or close to the client; for the Web-service namespace interface or file-system interface, this is the location of the parent directory</li> <li>▶ <b>\$client</b>: designates the location of the client where an operation is executed; A policy using the <code>\$client</code> location descriptor stores the replica on a storage service at the same location as the client that initiated the request.</li> <li>▶ <b>\$clientCreateLoc</b>: <code>\$clientCreateLoc</code> designates the location of the client where the object is/was created; a policy using the <code>\$clientCreateLoc</code> location descriptor stores the replica on a storage service at the same location as the client where the object is/was created.</li> <li>▶ <b>[rmglocations]</b>: A specific RMG location.</li> </ul> <p>Restrictions: You cannot combine the location_modifier <code>otherThan</code> with the location_place of <code>ANY</code>.</p> <p><b>Example:</b> <code>metadata[1][location_place]=\$client</code></p> |

Use the parameters in the following table to define the user data replicas. This part contains the following optional sections:

- ▶ **Customize**: If you enable `customize`, you can define how the storage service allocates new objects to the physical disks under its management. The `ssattrs_actions` and `ssattrs_placement` parameters are required when you enable customization.
- ▶ **Enable stripe**: Striping allows you to put blocks of data of the same replica on different nodes. Once you enable striping, you must specify the `stripe_number`, `stripe_size`, and `stripe_units` parameters.

- ▶ **Federate:** If you enable federation, you must also specify the `service_name` parameter. You can only specify the federation names that were defined by the SysAdmin.
- ▶ **Erasure Code:** Erasure coding provides data redundancy without the overhead of replication. Erasure Code divides an object into `m` data fragments and `k` coding fragments. Atmos supports the following fragmentation schemes: Data=9, Code=3 Data=10, Code=6. You must supply the following parameters: `alg=CRS`, `fragmentation=fragmentation1` for 9:3) or `fragmentation2` (for 10:6).

**NOTE:** You can define any number of data replicas, this is an array, and every element the array represents a single replica.. The `[x]` indicates the replica number. Replicas start at 1.

| Parameter                               | Description   |
|---|---|
| <code>spec[x][type]</code>              | Specifies the replica type. Valid values: <code>sync</code> , <code>async</code> . If there is more than one replica, then the value for at least one must be <code>sync</code> .<br><br><b>Example:</b> <code>spec [1] [type] =sync</code> |
| <code>spec[x][location_modifier]</code> | The location comparison. Valid values: <code>sameAs</code> (no replica customization), <code>otherThan</code> .<br><br><b>Example:</b> <code>spec [1] [location_modifier] =sameAs</code>  |
| <code>spec[x][location_place]</code>    | Specify the placement. Valid values: <code>ANY</code> (no replica customization), <code>\$client</code> , RMG locations.<br><br><b>Example:</b> <code>spec [1] [location_place] =%24client</code>   |
| <code>spec[x][enable_customize]</code>  | (Optional.) Specify if you want to specify the location and storage server attributes parameters:<br><br><b>Example:</b> <code>spec [1] [enable_customize] =on</code>   |

| Parameter                  | Description  |
|----------------------------|--|
| spec[x][ssattrs_placement] | <p>Use when customization is on. Specify the storage server placement method that defines how the storage service allocates new objects to the physical disks under its management. Options are:</p> <p><b>OPTIMAL</b> — Round robin: rotate requests equally among all available storage disks. This is the default.</p> <p><b>GREEN</b> — Only one or a few disks are active; the rest are spun down for energy conservation. When the active disks fill up, the spun-down disks are woken up and used.</p> <p><b>FAST</b> — Use striping by putting blocks of data of the same replica on different disks. This mode allows for a better aggregate throughput to the disks, by using multiple disk heads to access the same replica. This is like RAID striping.</p> <p><b>BALANCED</b> — Pick the disk with the most available space.</p> <p>The only value that can be specified here is what was specified during RMG configuration.</p> <p>In a VMWare installation, GREEN and FAST are not supported.</p> <p><b>Example:</b> <code>spec [1] [ssattrs_placement] =OPTIMAL</code></p>  |
| spec[x][ssattrs_actions]   | <p>Use when customization is on. Specify the data-at-rest services that are applied to replicas when they are stored. Valid values are:</p> <p><b>ANY</b> — Pick any storage server, regardless of how it is configured (e.g., whether it is configured for compression).</p> <p><b>NONE</b> — No services; normal writes and reads. This is the default.</p> <p><b>Checksum</b> — the storage server generates a checksum for all data saved on it. When the data is read, the checksum is verified. When the data is re-written, the old data is also verified.</p> <p><b>COMPRESSION</b> — Data is compressed on write and decompressed on read. This conserves disk space but introduces processing overhead on the server, so typically it has a negative impact on performance. Compression can be used successfully for objects that are written once and accessed rarely.</p> <p><b>DEDUPLICATION</b> — multiple copies of the data are removed, and only one copy of the data is maintained. This is intended to conserve storage capacity.</p> <p><b>COMPRESSION,DEDUPLICATION</b> — Combination of <b>COMPRESSION</b> and <b>DEDUPLICATION</b> can only be specified if those attributes are available at the specified location.</p> |

| Parameter                | Description  |
|--------------------------|--|
| spec[x][enable_stripe]   | <p>Optional. Specify if you want to enable striping. If it is, you must also specify the stripe_size integer, stripe_number integer, stripe_units.</p> <p>If you enable stripe, you can also enable customize, but you cannot enable flexout.</p> <p><b>Example:</b> <code>spec [1] [enable_stripe] =on</code></p> |
| spec[x][stripe_number]=  | <p>Use when striping is enabled. Specifies the number of nodes to stripe across.</p> <p><b>Example:</b> <code>spec [1] [stripe_number] =</code></p>  |
| spec[x][stripe_size]     | <p>Use when striping is enabled. Specify the amount of data written to each node.</p> <p><b>Example:</b> <code>spec [1] [stripe_size] =</code></p>   |
| spec[x][stripe_units]    | <p>Use when striping is enabled. Specify the stripe units. Valid values: B (no striping), KB, MB, GB, TB.</p> <p><b>Example:</b> <code>spec [1] [stripe_units] =B</code></p>   |
| spec[x][enable_flexout]  | <p>(Optional). Specify this parameter if you want to enable federation.</p> <p>When you enable flexout, you must specify the service_name parameter. If you enable flexout, you cannot also specify striping or customization.</p> <p><b>Example:</b> <code>spec [1] [enable_flexout] =on</code></p>               |
| spec[x][service_name]    | <p>Use when federation is enabled. Specify the service_name's valid value is the [Cloud Federations name list] in the system configuration.</p> <p><b>Example:</b> <code>spec [1] [service_name] =</code></p>  |
| spec[x][enable_erasure]= | <p>Optional. Specify if you want to enable erasure coding.</p> <p><b>Example:</b> <code>spec [1] [enable_erasure] =on</code></p>   |
| spec[x][alg]=CRS         | <p>Specifies the erasure coding algorithm. Value is always CRS.</p> <p><b>Example:</b> <code>spec [1] [alg] =CRS</code></p>  |
| spec[x][fragmentation]=  | <p>Use to specify the erasure coding fragmentation scheme. Values are fragmentation1 for 9:3 and fragmentation2 for 10:6.</p> <p><b>Example:</b> <code>spec [1] [fragmentation] =fragmentation2</code></p>   |

Use the parameters in the the following table to define the read access, retention rules, and deletion rules. Retention and deletion are optional values and must be enabled before you can specify any values.

| Parameter                            | Description  |
|--------------------------------------|--|
| <code>other[read_access]</code>      | <p>Specify the replicas to use for read access.</p> <p>Valid values:</p> <ul style="list-style-type: none"> <li>▶ <b>geographic</b>: Chooses the closest replica in geographic terms. This is the default.</li> <li>▶ <b>random</b>: Picks a replica at random.</li> </ul> <p><b>Example:</b> <code>other[read_access]=geographic</code></p> |
| <code>other[enable_retention]</code> | <p>(Optional.) Specify if you want to enable retention. The valid values are natural numbers like 0, 1, 2, and so on. There is no upper limit.</p> <p>If you specify both retention and deletion, the deletion length must be longer than retention length.</p> <p><b>Example:</b> <code>other[enable_retention]=on</code></p>               |
| <code>other[retention_year]</code>   | Specify the retention year   |
| <code>other[retention_month]</code>  | Specify the retention Month  |
| <code>other[retention_day]</code>    | Specify the retention day  |
| <code>other[retention_hour]</code>   | Specify the retention hour   |
| <code>other[retention_minute]</code> | Specify the retention minute   |
| <code>other[retention_second]</code> | Specify the retention second   |
| <code>other[enable_deletion]</code>  | <p>(Optional). Specify if you want to enable deletion. Valid values are natural number like 0, 1, 2, ... , and so on. There is no upper limit.</p> <p>If you specify both retention and deletion, the deletion length must be longer than retention length.</p> <p><b>Example:</b> <code>other[enable_deletion]=on</code></p>                |
| <code>other[deletion_year]</code>    | Specify the deletion year  |
| <code>other[deletion_month]</code>   | Specify the deletion month   |
| <code>other[deletion_day]</code>     | Specify the deletion day   |
| <code>other[deletion_hour]</code>    | Specify the deletion hour  |
| <code>other[deletion_minute]</code>  | Specify the deletion minute  |
| <code>other[deletion_second]</code>  | Specify the deletion second.   |

This example creates a policy specification called `t1`. It specifies one sync striped replica on storage servers with optimal layout.

|                        |   |
|------------------------|---|
| <b>Request Header</b>  | <pre> POST /tenant_admin/submit_create_spec HTTP/1.1 Accept: application/xml Content-Type: application/x-www-form-urlencoded Cookie: _gui_session_id=04a14f8c3dbd25426c5885e7808cfd84 Host: 10.5.116.244 Content-Length: 852 </pre>   |
| <b>Request Body</b>    | <pre> perform_action=create&amp;other[specname]=t1&amp;entry[spec_name]=&amp;&amp;new_spec=true&amp;replica_id =&amp;metadata[1][location_modifier]=sameAs&amp;metadata[1][location_place]=\$client&amp;spec[1][ type]=sync&amp;spec[1][enable_customize]=on&amp;spec[1][enable_stripe]=on&amp;spec[1][location_m odifier]=sameAs&amp;spec[1][location_place]=\$client&amp;spec[1][ssattrs_placement]=OPTIMAL&amp;s pec[1][ssattrs_actions]=NONE&amp;spec[1][stripe_number]=4&amp;spec[1][stripe_size]=40&amp;spec[1 ][stripe_units]=MB&amp;spec[1][service_name]=&amp;spec[1][alg]=CRS&amp;spec[1][fragmentation]=fr agmentation1&amp;other[read_access]=geographic&amp;other[retention_year]=&amp;other[retention_mo nth]=&amp;other[retention_day]=&amp;other[retention_hour]=&amp;other[retention_minute]=&amp;other[re tention_second]=&amp;other[deletion_year]=&amp;other[deletion_month]=&amp;other[deletion_day]=&amp;o ther[deletion_hour]=&amp;other[deletion_minute]=&amp;other[deletion_second]=&amp;operation_type= sync </pre> |
| <b>Response Header</b> | <pre> HTTP/1.1 200 OK Date: Wed, 16 Sep 2009 08:10:25 GMT Server: Mongrel 1.1.3 Status: 200 OK Cache-Control: no-cache Content-Type: application/xml; charset=utf-8 Content-Length: 20 Set-Cookie: _gui_session_id=04a14f8c3dbd25426c5885e7808cfd84; path=/ Connection: close </pre>  |
| <b>Response Body</b>   | <p>A successful request returns:</p> <pre>&lt;added&gt;true&lt;/added&gt;</pre>   |

## Delete a Policy Specification

|                           |  |
|---------------------------|--|
| <b>Description</b>        | <p>Deletes the specified policy. Returns a Boolean representing success (true) or failure (an error message) of the request. The HTTP request header must include the <code>Cookie</code> with the session ID (<code>_gui_session_id</code>) returned from a successful authentication.</p>  |
| <b>Required Role</b>      | TenantAdmin  |
| <b>HTTP Method</b>        | GET  |
| <b>URI</b>                | <code>/tenant_admin/submit_delete_spec</code>  |
| <b>Request Parameters</b> | <p>HTTP header:</p> <ul style="list-style-type: none"> <li>▷ <b>Cookie:</b> Set to the <code>_gui_session_id</code> returned by the authentication method.</li> </ul> <p>Querystring parameter:</p> <ul style="list-style-type: none"> <li>▷ <b>spec_name:</b> Specify the policy name.</li> <li>▷ <b>operation_type:</b> sync or async. Sync is the default.</li> </ul> |

**Request Header** GET /tenant\_admin/submit\_delete\_spec?spec\_name=SamplePolicy&operation\_type=sync  
HTTP/1.1  
Accept: application/xml  
Cookie: \_gui\_session\_id=72435064e18f97f0f18273bce644c41e  
Host: 10.5.116.244

**Request Body** None.

**Response Header** HTTP/1.1 200 OK  
Date: Wed, 16 Sep 2009 07:33:41 GMT  
Server: Mongrel 1.1.3  
Status: 200 OK  
Cache-Control: no-cache  
Content-Type: application/xml; charset=utf-8  
Content-Length: 110  
Set-Cookie: \_gui\_session\_id=72435064e18f97f0f18273bce644c41e; path=/  
Connection: close

**Response Body** A successful request returns:

```
<deleted>true</deleted>
```

An unsuccessful request returns an error message. The following is an example:

```
<deleted>Failed to delete placement policy. It's used by at least one policy selector  
or subtenant.</deleted>
```

## Get a Policy Specification

**Description** Returns an XML document containing the following information about the specified policy:

- ▶ Policy name
- ▶ metadata location modifier and metadata location
- ▶ The list of replicas and the replica attributes

The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role** TenantAdmin

**HTTP Method** GET

**URI** /tenant\_admin/get\_policy\_spec

**Request Parameters** HTTP header:

- ▶ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

Querystring parameter:

- ▷ **spec\_name**: Specify the policy name.

|                        |  |
|------------------------|--|
| <b>Request Header</b>  | GET /tenant_admin/get_policy_spec?spec_name=2LSand1RA HTTP/1.1<br>Accept: application/xml<br>Cookie: _gui_session_id=f25456ba05eea359566ab263f91e89ef<br>Host: 10.5.116.244  |
| <b>Request Body</b>    | None   |
| <b>Response Header</b> | HTTP/1.1 200 OK<br>Date: Wed, 16 Sep 2009 07:15:17 GMT<br>Server: Mongrel 1.1.3<br>Status: 200 OK<br>Cache-Control: no-cache<br>Content-Type: application/xml; charset=utf-8<br>Content-Length: 1935<br>Set-Cookie: _gui_session_id=f25456ba05eea359566ab263f91e89ef; path=/<br>Connection: close  |
| <b>Response Body</b>   | A successful request returns:<br><pre>&lt;policy&gt;   &lt;name&gt;2LSand1RA&lt;/name&gt;   &lt;metadata_location_modifier&gt;&lt;/metadata_location_modifier&gt;   &lt;metadata_location_place&gt;&lt;/metadata_location_place&gt;   &lt;replica_list&gt;     &lt;replica&gt;       &lt;id&gt;0&lt;/id&gt;       &lt;type&gt;sync&lt;/type&gt;       &lt;enable_customize&gt;on&lt;/enable_customize&gt;       &lt;location_modifier&gt;sameAs&lt;/location_modifier&gt;       &lt;location_place&gt;\$client&lt;/location_place&gt;       &lt;ssattrs_placement&gt;OPTIMAL&lt;/ssattrs_placement&gt;       &lt;ssattrs_actions&gt;NONE&lt;/ssattrs_actions&gt;       &lt;enable_striping&gt;&lt;/enable_striping&gt;       &lt;stripe_number&gt;&lt;/stripe_number&gt;       &lt;stripe_size&gt;&lt;/stripe_size&gt;       &lt;stripe_unit&gt;&lt;/stripe_unit&gt;       &lt;enable_flexout&gt;&lt;/enable_flexout&gt;       &lt;cloud_service&gt;&lt;/cloud_service&gt;     &lt;/replica&gt;     &lt;replica&gt;       &lt;id&gt;1&lt;/id&gt;       &lt;type&gt;sync&lt;/type&gt;       &lt;enable_customize&gt;on&lt;/enable_customize&gt;       &lt;location_modifier&gt;sameAs&lt;/location_modifier&gt;       &lt;location_place&gt;\$client&lt;/location_place&gt;       &lt;ssattrs_placement&gt;OPTIMAL&lt;/ssattrs_placement&gt;       &lt;ssattrs_actions&gt;NONE&lt;/ssattrs_actions&gt;       &lt;enable_striping&gt;&lt;/enable_striping&gt;       &lt;stripe_number&gt;&lt;/stripe_number&gt;       &lt;stripe_size&gt;&lt;/stripe_size&gt;       &lt;stripe_unit&gt;&lt;/stripe_unit&gt;       &lt;enable_flexout&gt;&lt;/enable_flexout&gt;       &lt;cloud_service&gt;&lt;/cloud_service&gt;     &lt;/replica&gt;     &lt;replica&gt;       &lt;id&gt;2&lt;/id&gt;       &lt;type&gt;async&lt;/type&gt;</pre> |

```

        <enable_customize>on</enable_customize>
        <location_modifier>otherThan</location_modifier>
        <location_place>$client</location_place>
        <ssattrs_placement>OPTIMAL</ssattrs_placement>
        <ssattrs_actions>NONE</ssattrs_actions>
        <enable_stripping></enable_stripping>
        <stripe_number></stripe_number>
        <stripe_size></stripe_size>
        <stripe_unit></stripe_unit>
        <enable_flexout></enable_flexout>
        <cloud_service></cloud_service>
    </replica>
</replica_list>
<read_access>geographic</read_access>
<enable_retention></enable_retention>
<enable_deletion></enable_deletion>
</policy>

```

## List Policy Specifications

**Description** Returns an XML document that contains general information about all policy specifications for the tenant. The XML document contains the following information:

- ▶ Policy Name
- ▶ List of replicas including replica type, storage mechanism, and storage location.

The HTTP request header must include the `Cookie` with the session ID (`_gui_session_id`) returned from a successful authentication.

**Required Role** TenantAdmin

**HTTP Method** GET

**URI** /tenant\_admin/list\_policy

**Request Parameters** HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

**Request Header** GET /tenant\_admin/list\_policy HTTP/1.1  
 Accept: application/xml  
 Cookie: \_gui\_session\_id=7403fa595c3065898f20045dd8dd67dc  
 Host: 10.5.116.244

**Request Body** None

**Response Header**

```
HTTP/1.1 200 OK
Date: Wed, 16 Sep 2009 07:23:04 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 2219
Set-Cookie: _gui_session_id=7403fa595c3065898f20045dd8dd67dc; path=/
Connection: close
```

**Response Body**

A successful request returns:

```
<policy_list>
  <policy>
    <name>2LSand1RA</name>
    <replica_list>
      <replica>
        <type>sync</type>
        <storage_mechanism>OPTIMAL</storage_mechanism>
        <location>$client</location>
      </replica>
      <replica>
        <type>sync</type>
        <storage_mechanism>OPTIMAL</storage_mechanism>
        <location>$client</location>
      </replica>
      <replica>
        <type>async</type>
        <storage_mechanism>OPTIMAL</storage_mechanism>
        <location>$client</location>
      </replica>
    </replica_list>
    <retention></retention>
    <deletion></deletion>
  </policy>
  <policy>
    <name>1LSand1RA</name>
    <replica_list>
      <replica>
        <type>sync</type>
        <storage_mechanism>OPTIMAL</storage_mechanism>
        <location>$client</location>
      </replica>
      <replica>
        <type>async</type>
        <storage_mechanism>OPTIMAL</storage_mechanism>
        <location>$client</location>
      </replica>
    </replica_list>
    <retention></retention>
    <deletion></deletion>
  </policy>
  <policy>
    <name>1LSand1LScomp</name>
    <replica_list>
```

```

    <replica>
      <type>sync</type>
      <storage_mechanism>OPTIMAL</storage_mechanism>
      <location>$client</location>
    </replica>
    <replica>
      <type>sync</type>
      <storage_mechanism>OPTIMAL</storage_mechanism>
      <location>$client</location>
    </replica>
  </replica_list>
  <retention></retention>
  <deletion></deletion>
</policy>
</policy_list>

```

## Modify a Policy Specification

**Description** Modifies the specified policy. This is a replacement of the existing policy with the new values.

**Required Role** TenantAdmin

**HTTP Method** POST

**URI** POST /tenant\_admin/submit\_modify\_spec

**Request Parameters** HTTP header:

- ▷ **Cookie:** Set to the `_gui_session_id` returned by the authentication method.

HTTP body:

The following table describes the general parameters.

| Parameter        | Description  |
|------------------|--|
| perform_action   | Specify <code>create</code> (for create operations) or <code>modify</code> (for modify operations).<br><br><b>Example:</b> <code>perform_action=create</code>                                |
| other[specname]  | Specify the name of the policy to create or modify. The name can be any string, but it cannot include special characters.<br><br><b>Example:</b> <code>other [specname] =samplepolicy</code> |
| entry[spec_name] | Not used. Supply the parameter with no value.  |

| Parameter  | Description   |
|------------|---|
| new_spec   | Specify true for all operations.<br><br><b>Example:</b> new_spec=true |
| replica_id | Not used. Supply the parameter with no value.                         |

Use the parameters in the following table to define metadata should be treated:

| Parameter                      | Description  |
|--------------------------------|--|
| metadata[x][location_modifier] | <p>Specifies the matching criteria for the location_place. [x] indicates replica number. For metadata the value is always 1.</p> <p>Valid values are: sameAs (no replica customization), and otherThan.</p> <p>You cannot combine the location_modifier otherThan with the location_place of ANY.</p> <p><b>Example:</b> metadata[1][location_modifier]=sameAs</p>   |
| metadata[x][location_place]    | <p>Specifies the location where Atmos should create new files and directories. [x] indicates replica number. For metadata the value is always 1.</p> <p>Valid values:</p> <ul style="list-style-type: none"> <li>▶ <b>ANY</b> (the default location): For the Web-service object interface, this is a location at or close to the client; for the Web-service namespace interface or file-system interface, this is the location of the parent directory</li> <li>▶ <b>\$client</b>: designates the location of the client where an operation is executed; A policy using the \$client location descriptor stores the replica on a storage service at the same location as the client that initiated the request.</li> <li>▶ <b>\$clientCreateLoc</b>: \$clientCreateLoc designates the location of the client where the object is/was created; a policy using the \$clientCreateLoc location descriptor stores the replica on a storage service at the same location as the client where the object is/was created.</li> <li>▶ <b>[rmglocations]</b>: A specific RMG location.</li> </ul> <p>Restrictions: You cannot combine the location_modifier otherThan with the location_place of ANY.</p> <p><b>Example:</b> metadata[1][location_place]=\$client</p> |

Use the parameters in the following table to define the user data replicas. This part contains the following optional sections:

- ▶ **Customize:** If you enable `customize`, you can define how the storage service allocates new objects to the physical disks under its management. The `ssattrs_actions` and `ssattrs_placement` parameters are required when you enable customization.
- ▶ **Enable stripe:** Striping allows you to put blocks of data of the same replica on different nodes. Once you enable striping, you must specify the `stripe_number`, `stripe_size`, and `stripe_units` parameters.
- ▶ **Enable Federation:** If you enable federation, you must also specify the `service_name` parameter. You can only specify the federation names that were defined by the SysAdmin.
- ▶ **Erasure Code:** Erasure coding provides data redundancy without the overhead of replication. Erasure Code divides an object into `m` data fragments and `k` coding fragments. Atmos supports the following fragmentation schemes: Data=9, Code=3 Data=10, Code=6. You must supply the following parameters: `alg=CRS`, `fragmentation=fragmentation1` for 9:3) or `fragmentation2` (for 10:6).

**NOTE:** You can define any number of data replicas, this is an array, and every element the array represents a single replica.. The `[x]` indicates the replica number. Replicas start at 1.

| Parameter                               | Description   |
|---|---|
| <code>spec[x][type]</code>              | Specifies the replica type. Valid values: <code>sync</code> , <code>async</code> . If there is more than one replica, then the value for at least one must be <code>sync</code> .<br><br><b>Example:</b> <code>spec [1] [type] =sync</code> |
| <code>spec[x][location_modifier]</code> | The location comparison. Valid values: <code>sameAs</code> (no replica customization), <code>otherThan</code> .<br><br><b>Example:</b> <code>spec [1] [location_modifier] =sameAs</code>  |
| <code>spec[x][location_place]</code>    | Specify the placement. Valid values: <code>ANY</code> (no replica customization), <code>\$client</code> , RMG locations.<br><br><b>Example:</b> <code>spec [1] [location_place] =%24client</code>   |
| <code>spec[x][enable_customize]</code>  | (Optional.) Specify if you want to specify the location and storage server attributes parameters:<br><br><b>Example:</b> <code>spec [1] [enable_customize] =on</code>   |

| Parameter                  | Description  |
|----------------------------|--|
| spec[x][ssattrs_placement] | <p>Use when customization is on. Specify the storage server placement method that defines how the storage service allocates new objects to the physical disks under its management. Options are:</p> <p><b>OPTIMAL</b> — Round robin: rotate requests equally among all available storage disks. This is the default.</p> <p><b>GREEN</b> — Only one or a few disks are active; the rest are spun down for energy conservation. When the active disks fill up, the spun-down disks are woken up and used.</p> <p><b>FAST</b> — Use striping by putting blocks of data of the same replica on different disks. This mode allows for a better aggregate throughput to the disks, by using multiple disk heads to access the same replica. This is like RAID striping.</p> <p><b>BALANCED</b> — Pick the disk with the most available space.</p> <p>The only value that can be specified here is what was specified during RMG configuration.</p> <p>In a VMWare installation, GREEN and FAST are not supported.</p> <p><b>Example:</b> <code>spec [1] [ssattrs_placement]=OPTIMAL</code></p>   |
| spec[x][ssattrs_actions]   | <p>Use when customization is on. Specify the data-at-rest services that are applied to replicas when they are stored. Valid values are:</p> <p><b>ANY</b> — Pick any storage server, regardless of how it is configured (e.g., whether it is configured for compression).</p> <p><b>NONE</b> — No services; normal writes and reads. This is the default.</p> <p><b>Checksum</b> — the storage server generates a checksum for all data saved on it. When the data is read, the checksum is verified. When the data is re-written, the old data is also verified.</p> <p><b>COMPRESSION</b> — Data is compressed on write and decompressed on read. This conserves disk space but introduces processing overhead on the server, so typically it has a negative impact on performance. Compression can be used successfully for objects that are written once and accessed rarely.</p> <p><b>DEDUPLICATION</b> — multiple copies of the data are removed, and only one copy of the data is maintained. This is intended to conserve storage capacity.</p> <p><b>COMPRESSION,DEDUPLICATION</b> — Combination of <b>COMPRESSION</b> and <b>DEDUPLICATION</b> can only be specified if those attributes are available at the specified location.</p> <p><b>Example:</b> <code>spec [1] [ssattrs_actions]=NONE</code></p> |

| Parameter                | Description  |
|--------------------------|--|
| spec[x][enable_stripe]   | <p>(Optional.) Specify if you want to enable striping.</p> <p>If you enable stripe, you can also enable customize, but you cannot enable flexout.</p> <p><b>Example:</b> <code>&amp;spec [1] [enable_stripe] =on</code></p>  |
| spec[x][stripe_number]=  | <p>Use when striping is enabled. Specifies the number of nodes to stripe across.</p> <p><b>Example:</b> <code>spec [1] [stripe_number] =</code></p>  |
| spec[x][stripe_size]     | <p>Use when striping is enabled. Specify the amount of data written to each node.</p> <p><b>Example:</b> <code>spec [1] [stripe_size] =</code></p>   |
| spec[x][stripe_units]    | <p>Use when striping is enabled. Specify the stripe units. Valid values: B (no striping), KB, MB, GB, TB.</p> <p><b>Example:</b> <code>spec [1] [stripe_units] =B</code></p>   |
| spec[x][enable_flexout]  | <p>(Optional). Specify if you want to enable federation.</p> <p>When you enable flexout, you must specify the <code>service_name</code> parameter. If you enable flexout, you cannot also specify striping or customization.</p> <p><b>Example:</b> <code>spec [1] [enable_flexout] =on</code></p> |
| spec[x][service_name]    | <p>Use when federation is enabled. Specify the <code>service_name</code>'s valid value is the [Cloud Federations name list] in the system configuration.</p> <p><b>Example:</b> <code>spec [1] [service_name] =</code></p>   |
| spec[x][enable_erasure]= | <p>Optional. Specify if you want to enable erasure coding.</p> <p><b>Example:</b> <code>spec [1] [enable_erasure] =on</code></p>   |
| spec[x][alg]=CRS         | <p>Specifies the erasure coding algorithm. Value is always CRS.</p> <p><b>Example:</b> <code>spec [1] [alg] =CRS</code></p>  |
| spec[x][fragmentation]=  | <p>Use to specify the erasure coding fragmentation scheme. Values are <code>fragmentation1</code> for 9:3 and <code>fragmentation2</code> for 10:6.</p> <p><b>Example:</b> <code>spec [1] [fragmentation] =fragmentation2</code></p>   |

Use the parameters in the the following table to define the read access, retention rules, and deletion rules. Retention and deletion are optional values and must be enabled before you can specify any values.

| Parameter                            | Description  |
|--------------------------------------|--|
| <code>other[read_access]</code>      | <p>Specify the replicas to use for read access.</p> <p>Valid values:</p> <ul style="list-style-type: none"> <li>▶ <b>geographic</b>: Chooses the closest replica in geographic terms. This is the default.</li> <li>▶ <b>random</b>: Picks a replica at random.</li> </ul> <p><b>Example:</b> <code>other[read_access]=geographic</code></p> |
| <code>other[enable_retention]</code> | <p>(Optional.) Specify if you want to enable retention. The valid values are natural numbers like 0, 1, 2, and so on. There is no upper limit.</p> <p>If you specify both retention and deletion, the deletion length must be longer than retention length.</p> <p><b>Example:</b> <code>other[enable_retention]=on</code></p>               |
| <code>other[retention_year]</code>   | <p>Specify the retention year</p> <p><b>Example:</b> <code>other[retention_year]=</code></p>   |
| <code>other[retention_month]</code>  | <p>Specify the retention month.</p> <p><b>Example:</b> <code>other[retention_month]=</code></p>  |
| <code>other[retention_day]</code>    | <p>Specify the retention day.</p> <p><b>Example:</b> <code>other[retention_day]=</code></p>  |
| <code>other[retention_hour]</code>   | <p>Specify the retention hour.</p> <p><b>Example:</b> <code>other[retention_hour]=4</code></p>   |
| <code>other[retention_minute]</code> | <p>Specify the retention minute.</p> <p><b>Example:</b> <code>other[retention_minute]=</code></p>  |
| <code>other[retention_second]</code> | <p>Specify the retention second.</p> <p><b>Example:</b> <code>other[retention_second]=</code></p>  |
| <code>other[enable_deletion]</code>  | <p>(Optional). Specify if you want to enable deletion. Valid values are natural numbers like 0, 1, 2, ... , and so on. There is no upper limit.</p> <p>If you specify both retention and deletion, the deletion length must be longer than retention length.</p> <p><b>Example:</b> <code>other[enable_deletion]=on</code></p>               |
| <code>other[deletion_year]</code>    | <p>Specify the deletion year.</p> <p><b>Example:</b> <code>other[deletion_year]=</code></p>  |

| Parameter              | Description   |
|------------------------|---|
| other[deletion_month]  | Specify the deletion month.<br><br><b>Example:</b> other[deletion_month]=   |
| other[deletion_day]    | Specify the deletion day.<br><br><b>Example:</b> other[deletion_day]        |
| other[deletion_hour]   | Specify the deletion hour<br><br><b>Example:</b> other[deletion_hour]=      |
| other[deletion_minute] | Specify the deletion minute.<br><br><b>Example:</b> other[deletion_minute]= |
| other[deletion_second] | Specify the deletion second.<br><br><b>Example:</b> other[deletion_second]= |

The following example modifies an existing policy named t1. It specifies one sync, striped replica on storage servers with optimal layout.

#### Request Header

```
POST /tenant_admin/submit_modify_spec HTTP/1.1
Accept: application/xml
Accept-Type: application/x-www-form-urlencoded
Cookie: _gui_session_id=b607c4a5a6d616c2ec22bbc959e7015b
Host: 10.5.116.244
Content-Length: 859
```

#### Request Body

```
perform_action=modify&other[specname]=t1&entry[spec_name]=&&new_spec=true&replica_id
=&metadata[1][location_modifier]=sameAs&metadata[1][location_place]=$client&spec[1][
type]=sync&spec[1][enable_customize]=on&spec[1][enable_stripe]=on&spec[1][location_m
odifier]=sameAs&spec[1][location_place]=$client&spec[1][ssattrs_placement]=OPTIMAL&s
pec[1][ssattrs_actions]=NONE&spec[1][stripe_number]=5&spec[1][stripe_size]=55&spec[1
][stripe_units]=MB&spec[1][service_name]=&spec[1][alg]=CRS&spec[1][fragmentation]=fr
agmentation1&other[read_access]=geographic&other[retention_year]=&other[retention_mo
nth]=&other[retention_day]=&other[retention_hour]=&other[retention_minute]=&other[re
tention_second]=&other[deletion_year]=&other[deletion_month]=&other[deletion_day]=&o
ther[deletion_hour]=&other[deletion_minute]=&other[deletion_second]=&operation_type=
sync
```

#### Response Header

```
HTTP/1.1 200 OK
Date: Wed, 16 Sep 2009 08:34:46 GMT
Server: Mongrel 1.1.3
Status: 200 OK
Cache-Control: no-cache
Content-Type: application/xml; charset=utf-8
Content-Length: 24
Set-Cookie: _gui_session_id=b607c4a5a6d616c2ec22bbc959e7015b; path=/
Connection: close
```

#### Response Body

A successful request returns:

```
<cleared>true</cleared>
```

An unsuccessful request returns an error message like this:

```
<cleared>Policy not found for specified ID</cleared>
```



**PART II:**

**ATMOS SYSTEM MONITORING REST  
API**

# 7

## System Monitoring API

This chapter describes the EMC® Atmos™ System Monitoring API. It includes the following topics:

- ▶ [About the System Monitoring API](#)
- ▶ [API Reference](#)
- ▶ [Error Responses](#)

---

### About the System Monitoring API

The Atmos System Monitoring API is REST-based. It returns XML. You can use this API to:

- ▶ Discover and monitor the set of RMGs and nodes under the management of a single user in the SysAdmin role. For example, you can
  - ▷ Get the list of RMGs, and for each RMG, the set of nodes.
  - ▷ Determine how many nodes are running or down.
  - ▷ View the average load for an RMG for 1, 5, and 15 minute intervals.
- ▶ Retrieve Web Service statistics for a specific node.
- ▶ Monitor storage consumption for a tenant, subtenant, or UID.

### Authentication, Data Integrity, and Privacy

To perform any of the operations in this API, the requests must include authentication headers that pass in the credentials of the user making the request. Each operation requires that the authenticating user be in a specific role. Each request also requires the `x-atmos-authtype` whose value must always be the string `password`.

Because these operations are related to system management, Atmos requires that the request be made using HTTPS on port 443, or the operation fails. For example:

```
https://<ipaddress>:443/sysmgmt/rmgs
```

You can perform these API operations only on the master node of an installation segment. If you attempt the operation on other nodes, you receive an authentication error.

---

## API Reference

- ▶ [RMG List](#)
- ▶ [RMG Info](#)
- ▶ [Nodes List](#)
- ▶ [Nodes info](#)
- ▶ [Web Service Statistics](#)
- ▶ [Atmos Storage Consumption](#)

### RMG List

|                           |  |
|---------------------------|--|
| <b>Description</b>        | Returns an XML document that provides the complete list of RMGs managed by the authenticated user in the SysAdmin role. The XML document provides the following details on each RMG: local time, number of CPUs, number of nodes that are up and the number that are down, and the average load for 1, 5, and 15 minute intervals. |
| <b>Required Role</b>      | SysAdmin   |
| <b>HTTP Method</b>        | GET  |
| <b>URI</b>                | <code>/sysmgmt/rmgs</code>   |
| <b>Request Parameters</b> | <ul style="list-style-type: none"><li>▶ <b>x-atmos-systemadmin</b>: Specify the SysAdmin username. (Required.)</li><li>▶ <b>x-atmos-systemadminpassword</b>: Specify the SysAdmin password. (Required.)</li><li>▶ <b>x-atmos-authtype</b>: Specify the string <code>password</code>.</li></ul>                                     |
| <b>Request Header</b>     | <pre>GET /sysmgmt/rmgs HTTP/1.1 accept: */* date: Thu, 12 Nov 2009 09:59:54 GMT x-atmos-systemadminpassword: password x-atmos-systemadmin: SysAdmin1 x-atmos-authtype: password</pre>  |
| <b>Request Body</b>       | None   |

**Response Header**

```

HTTP/1.1 200 OK
Connection: close
Date: Thu, 12 Nov 2009 10:00:00 GMT
Set-Cookie: _gui_session_id=afa8a3848c457bff0f8cddd6ebff2b53; path=/
Status: 200 OK
x-atoms-sysmgmt-version: 1.0.0
Cache-Control: no-cache
Server: Mongrel 1.1.3
Content-Type: application/xml; charset=utf-8
Content-Length: 328

```

**Response Body**

```

<?xml version="1.0" encoding="UTF-8"?>
<rmgList>
  <rmg name="Boston01">
    <name>Boston01</name>
    <localTime>Thu Nov 12 09:59:59 +0000 2009</localTime>
    <nodesUp>4</nodesUp>
    <nodesDown>0</nodesDown>
    <avgLoad15>110.25</avgLoad15>
    <avgLoad5>111.25</avgLoad5>
    <avgLoad1>104.0</avgLoad1>
  </rmg>
</rmgList>

```

The following table describes the elements of the response body.

| Element                       | Description   |
|-------------------------------|---|
| rmg                           | Container element for individual RMG.   |
| name                          | Specifies the RMG's name.   |
| localTime                     | Specifies the local time for the RMG.   |
| nodesUp/nodesDown             | Specifies the number of nodes in the RMG that are running and that are down.                |
| avgLoad15, avgLoad5, avgLoad1 | Measures how busy the Atmos services on the RMG are for one, five, and 15 minute intervals. |

**RMG Info****Description**

Returns information about a specific RMG.

**Required Role**

SysAdmin

**HTTP Method**

GET

**URI**

/sysmgmt/rmgs/<rmgName>

**Request Parameters**

- ▶ **x-atmos-systemadmin**: Specify the SysAdmin username. (Required.)
- ▶ **x-atmos-systemadminpassword**: Specify the SysAdmin password. (Required.)

- ▶ **x-atmos-authtype**: Specify the string password.

**Request Header**

```
Header: GET /sysmgmt/rmgs/Boston01 HTTP/1.1
accept: */*
date: Thu, 12 Nov 2009 10:01:29 GMT
x-atmos-systemadminpassword: password
x-atmos-systemadmin: SysAdmin1
x-atmos-authtype: password
```

**Request Body**

None

**Successful Response Header**

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 12 Nov 2009 10:01:33 GMT
Set-Cookie: _gui_session_id=82c33d236af1dc95ed6b5335dc714ec1; path=/
Status: 200 OK
x-atoms-sysmgmt-version: 1.0.0
Cache-Control: no-cache
Server: Mongrel 1.1.3
Content-Type: application/xml; charset=utf-8
Content-Length: 287
```

**Successful Response Body**

A successful response returns the following XML document:

```
<?xml version="1.0" encoding="UTF-8"?>

<rmg name="Boston01">
  <name>Boston01</name>
  <localTime>Thu Nov 12 10:01:28 +0000 2009</localTime>
  <nodesUp>4</nodesUp>
  <nodesDown>0</nodesDown>
  <avgLoad15>108.5</avgLoad15>
  <avgLoad5>107.75</avgLoad5>
  <avgLoad1>99.0</avgLoad1>
</rmg>
```

The following table describes the elements of the response body:

| Element                       | Description   |
|-------------------------------|---|
| rmg                           | Container element for individual RMG.   |
| name                          | Specifies the RMG's name  |
| localTime                     | Specifies the local time for the RMG.   |
| nodesUp/nodesDown             | Specifies the number of nodes that are running, and the number that are not running.        |
| avgLoad15, avgLoad5, avgLoad1 | Measures how busy the Atmos services on the RMG are for one, five, and 15 minute intervals. |

## Nodes List

|                                   |  |
|-----------------------------------|--|
| <b>Description</b>                | Returns the list of nodes for the specified RMG. For each node, it lists the node location, and whether it is up or down.  |
| <b>Required Role</b>              | SysAdmin   |
| <b>HTTP Method</b>                | GET  |
| <b>URI</b>                        | /sysmgmt/rmgs/<rmgName>/nodes  |
| <b>Request Parameters</b>         | <ul style="list-style-type: none"><li>▶ <b>x-atmos-systemadmin</b>: Specify the SysAdmin username. (Required.)</li><li>▶ <b>x-atmos-systemadminpassword</b>: Specify the SysAdmin password. (Required.)</li><li>▶ <b>x-atmos-authtype</b>: Specify the string password.</li></ul>  |
| <b>Request Header</b>             | <pre>GET /sysmgmt/rmgs/Boston01/nodes HTTP/1.1 accept: */* date: Thu, 12 Nov 2009 10:21:56 GMT x-atmos-systemadminpassword: password x-atmos-systemadmin: SysAdmin1 x-atmos-authtype: password</pre>   |
| <b>Request Body</b>               | None   |
| <b>Successful Response Header</b> | <pre>HTTP/1.1 200 OK Connection: close Date: Thu, 12 Nov 2009 10:22:01 GMT Set-Cookie: _gui_session_id=63da47f6abe6c4eb21e12f2328baca43; path=/ Status: 200 OK x-atoms-sysmgmt-version: 1.0.0 Cache-Control: no-cache Server: Mongrel 1.1.3 Content-Type: application/xml; charset=utf-8 Content-Length: 326</pre>   |
| <b>Successful Response Body</b>   | <pre>&lt;?xml version="1.0" encoding="UTF-8"?&gt; &lt;nodeList&gt;   &lt;node name="IS01-001" up="true" location="BostonFl1"&gt;&lt;/node&gt;   &lt;node name="IS01-002" up="true" location="BostonFl1"&gt;&lt;/node&gt;   &lt;node name="IS01-003" up="true" location="BostonFl1"&gt;&lt;/node&gt;   &lt;node name="IS01-004" up="true" location="BostonFl1"&gt;&lt;/node&gt; &lt;/nodeList&gt;</pre> |

The following table describes the elements of the response body:

| Element | Description                               |
|---------|---|
| node    | Container element for an individual node. |
| name    | Specifies the node's name.                |

| Element  | Description  |
|----------|--|
| location | Specifies the node's location.                                       |
| up       | Specifies true if the node is running, or false if the node is down. |

## Nodes info

**Description** Returns an XML document that provides details about the specified node. The details include both configuration (such as machine type, atmos version, and number of CPUs) as well as metrics for the node (such as number of processes, swap memory free and used, and so on).

**Required Role** SysAdmin

**HTTP Method** GET

**URI** /sysmgmt/rmg/<rmgName>/nodes/<nodeName>

- Request Parameters**
- ▶ **x-atmos-systemadmin:** Specify the SysAdmin username. (Required.)
  - ▶ **x-atmos-systemadminpassword:** Specify the SysAdmin password. (Required.)
  - ▶ **x-atmos-authtype:** Specify the string password.

**Request Header**

```
GET /sysmgmt/rmgs/Boston01/nodes/IS01-001 HTTP/1.1
accept: */*
date: Thu, 12 Nov 2009 10:24:07 GMT
x-atmos-systemadminpassword: password
x-atmos-systemadmin: SysAdmin1
x-atmos-authtype: password
```

**Successful Response Header**

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 12 Nov 2009 10:24:12 GMT
Set-Cookie: _gui_session_id=d29496c84a915be48bafeed9e051ae17; path=/
Status: 200 OK
x-atoms-sysmgmt-version: 1.0.0
Cache-Control: no-cache
Server: Mongrel 1.1.3
Content-Type: application/xml; charset=utf-8
Content-Length: 638
```

## Successful Response Body

```
<?xml version="1.0" encoding="UTF-8"?>
<node name="IS01-001" up="true" location="BostonFl1">
  <machineType>x86_64</machineType>
  <operationSystem>Linux</operationSystem>
  <atmosVersion>1.2.5</atmosVersion>
  <numberOfcpu>1</numberOfcpu>
  <cpuSpeed>1995</cpuSpeed>
  <totalMemory>1027372.000</totalMemory>
  <freeMemory>38784.000</freeMemory>
  <totalDiskSpace>6.231</totalDiskSpace>
  <numberOfprocesses>531</numberOfprocesses>
  <cpuSystemUsage>19.9</cpuSystemUsage>
  <cpuUserUsage>15.9</cpuUserUsage>
  <cpuIdle>64.2</cpuIdle>
  <totalSwapMemory>2104504.000</totalSwapMemory>
  <freeSwapMemory>2103008.000</freeSwapMemory>
</node>
```

The unit of measure for the response elements are defined below:

- ▶ `<cpuSpeed>`—MHz
- ▶ `<totalMemory>`—Bytes
- ▶ `<freeMemory>`—Bytes
- ▶ `<totalDiskSpace>`—GB
- ▶ `<cpuSystemUsage>`—a percentage
- ▶ `<cpuUserUsage>`—a percentage
- ▶ `<cpuIdle>`—a percentage
- ▶ `<totalSwapMemory>`—Bytes
- ▶ `<freeSwapMemory>`—Bytes

## Web Service Statistics

|                           |   |
|---------------------------|---|
| <b>Description</b>        | Use this operation to obtain basic node-level web service statistics.   |
| <b>Required Role</b>      | SysAdmin  |
| <b>HTTP Method</b>        | GET   |
| <b>URI</b>                | <code>/sysmgmt/rmg/&lt;rmgName&gt;/&lt;nodes&gt;/&lt;nodeName&gt;/wsstats</code>  |
| <b>Request Parameters</b> | <ul style="list-style-type: none"><li>▶ <b>x-atmos-systemadmin</b>: Specify the SysAdmin username. (Required.)</li><li>▶ <b>x-atmos-systemadminpassword</b>: Specify the SysAdmin password. (Required.)</li><li>▶ <b>x-atmos-authtype</b>: Specify the string password.</li></ul> |

**Request Header**

```
GET /sysmgmt/rmgs/Boston01/nodes/IS01-001/wsstats HTTP/1.1
accept: */*
date: Fri, 13 Nov 2009 09:26:47 GMT
x-atmos-systemadminpassword: password
x-atmos-systemadmin: SysAdmin1
x-atmos-authtype: password
```

**Request Body**

None

**Successful Response Header**

```
HTTP/1.1 200 OK
Connection: close
Date: Tue, 27 Oct 2009 03:33:05 GMT
Set-Cookie: _gui_session_id=96e1e72e9e7df2d7f151cfd4e65cec61; path=/
Status: 200 OK
x-atoms-sysmgmt-version: 1.0.0
Cache-Control: no-cache
Server: Mongrel 1.1.3
Content-Type: application/xml; charset=utf-8
Content-Length: 587
```

**Successful Response Body**

```
<?xml version='1.0' encoding='UTF-8'?>
<WSStat>
  <Node name='is01-001' />
  <ReadsPerSec>0.00</ReadsPerSec>
  <WritesPerSec>0.00</WritesPerSec>
  <DeletesPerSec>0.00</DeletesPerSec>
  <TransPerSec>0.00</TransPerSec>
  <MeanReadLatencyMS>6.723</MeanReadLatencyMS>
  <MeanWriteLatencyMS>214.721</MeanWriteLatencyMS>
  <MeanLatencyMS>110.722</MeanLatencyMS>
  <Reads>3</Reads>
  <Writes>3</Writes>
  <Deletes>0</Deletes>
  <Total>6</Total>
  <ReadLatencyMS>20</ReadLatencyMS>
  <WriteLatencyMS>644</WriteLatencyMS>
  <DeleteLatencyMS>0</DeleteLatencyMS>
  <UptimeMS>108302784</UptimeMS>
</WSStat>
```

- **Reads, Writes, and Deletes** are defined as follows:

|                | <b>REST</b>               | <b>SOAP</b>                                   |
|----------------|---------------------------|---|
| <b>Reads</b>   | GET and HEAD transactions | List, Get, and Read transactions              |
| <b>Writes</b>  | PUT and POST transactions | Create, Set, Update, and Version transactions |
| <b>Deletes</b> | DELETE transactions       | Delete transactions                           |

- **Total** is the sum of reads, writes, and deletes.
- The per-second (**PerSec**) rate for each of the three transaction types is calculated as the count divided by the total uptime since the last server restart.
- **Mean latency** is the sum of the individual latencies reported at logging time, divided by the total number of transactions. There also are latencies calculated by operation type in a similar manner.

- ▶ Separately, the total accumulated latency is reported for each operation (**ReadLatencyMS**, **WriteLatencyMS**, and **DeleteLatencyMS**) and for the uptime (**UptimeLatencyMS**), in milliseconds. This allows you to sample at any desired frequency and calculate latencies for that interval.

## Atmos Storage Consumption

|  |  |
|--|--|
| <b>Description</b>                       | Provides information about the number of objects and the total object size (object size and metadata size) for a specified tenant, subtenant, or UID.  |
| <b>Required Role</b>                     | TenantAdmin or SubtenantAdmin  |
| <b>HTTP Method</b>                       | GET  |
| <b>URI for Tenant Metrics</b>            | <code>/sysmgmt/tenants/&lt;tenantName&gt;/scMetrics</code>   |
| <b>URI for Subtenant Metrics</b>         | <code>/sysmgmt/tenants/&lt;tenantName&gt;/&lt;subTenantName&gt;/scMetrics</code>   |
| <b>URI for UID Metrics</b>               | <code>/sysmgmt/tenants/&lt;tenantName&gt;/&lt;subTenantName&gt;/&lt;UID&gt;/scMetrics</code>   |
| <b>Request Parameters</b>                | <ul style="list-style-type: none"> <li>▶ <b>x-atmos-tenantadmin</b>: Specify the TenantAdmin username. (Required.)</li> <li>▶ <b>x-atmos-tenantadminpassword</b>: Specify the TenantAdmin's password. (Required.)</li> <li>▶ <b>x-atmos-authtype</b>: Specify the string password.</li> </ul>                      |
| <b>Request Header for Tenant Metrics</b> | <pre>GET /sysmgmt/tenants/Tenant1/scMetrics HTTP/1.1 accept: */* date: Thu, 12 Nov 2009 10:04:11 GMT x-atmos-tenantadmin: Tenant1Admin x-atmos-tenantadminpassword: password x-atmos-authtype: password</pre>  |
| <b>Request Body</b>                      | None   |
| <b>Response Header</b>                   | <pre>HTTP/1.1 200 OK Connection: close Date: Thu, 12 Nov 2009 10:04:15 GMT Set-Cookie: _gui_session_id=2e336848d437fdafd9704b3157aeaff3; path=/ Status: 200 OK x-atoms-sysmgmt-version: 1.0.0 Cache-Control: no-cache Server: Mongrel 1.1.3 Content-Type: application/xml; charset=utf-8 Content-Length: 203</pre> |

## Response Body for Tenant Metrics

```
<?xml version='1.0' encoding='UTF-8'?>
<scMetrics>
  <objCount>5</objCount>
  <size>9845</size>
  <realSize>3306</realSize>
  <metadatasize>10434</metadatasize>
  <totalSize>13740</totalSize>
</scMetrics>
```

| Element      | Description   |
|--------------|---|
| objCount     | The number of objects stored for the specified tenant, subtenant, or UID.   |
| size         | The object size (not including replicas or metadata) for the specified tenant, subtenant, or UID. The default size for a directory is 4k. |
| realSize     | Derived from the size element multiplied by the number of replicas defined for the policy. Replica counts for directories are 0.          |
| metadatasize | The sum of the size of the XML description for user metadata, system metadata, policy description, LSO layout, and so on.                 |
| totalSize    | The sum of the realSize and the metadatasize elements.  |

## Error Responses

An unsuccessful response returns an XML document containing one of the HTTP status codes described in the table below.

### Failed Response Header

```
HTTP/1.1 #{statusCode}
Date: date
Content-Type: type
Content-Length: length
Connection: close
Server: mongrel
x-atmos-sysmgmt-version: #{version}
<?xml version='1.0' encoding='UTF-8'?>
```

### Failed Response Body

```
<errorResponse>
  <errorCode>#{errorCode}</errorCode>
  <errorMessage>#{error message}</errorMessage>
</errorResponse>
```

### HTTP Status Codes

| HTTP Status Code | Error Message  |
|------------------|--|
| 401              | Unauthorized. The client did not provide the correct tenant Admin name and password.             |
| 404              | Name not found.  |
| 500              | Server internal error. The error message in the response body provides the details of the error. |