Cisco Cloud Object Storage Release 2.1.1
API Guide

April 2015

Cisco Systems, Inc.
www.cisco.com

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco website at www.cisco.com/go/offices.
CONTENTS

Preface vii
Audience vii
Document Organization vii
Document Conventions viii
Related Publications ix
Obtaining Documentation and Submitting a Service Request ix

CHAPTER 1
Introduction to COS and COS APIs 1-1
Product Description 1-1
COS and MOS 1-1
Components 1-1
API Features 1-2
Overview 1-2
Service Manager API 1-3
Swauth API 1-3
Swift Object Store API 1-4
Restrictions and Limitations 1-5

CHAPTER 2
Service Manager API 2-1
Listing the Region 2-1
Listing DNS Servers 2-1
  Listing all DNS Servers 2-1
  Listing One DNS Server 2-2
Listing and Modifying NTP Servers 2-2
  Listing all NTP Servers 2-2
  Listing one NTP Server 2-2
  Updating an NTP Server 2-3
Listing DNS Forwarders 2-3
  Listing DNS Forwarders 2-3
  Listing One DNS Forwarder 2-4
Listing, Creating, Deleting, and Modifying Zones 2-4
  Listing All Zones 2-4
Listing One Zone 2-4
Creating a Zone 2-4
Deleting a Zone 2-5
Updating a Zone 2-5

Listing, Creating, Deleting, and Modifying Networks 2-5
Listing the Networks 2-5
Listing One Network 2-6
Creating a Network 2-6
Deleting a Network 2-7
Updating a Network 2-7

Listing, Creating, Deleting, and Modifying IP Pools 2-7
Listing the IP Pools 2-7
Listing One IP Pool 2-8
Creating an IP Pool 2-8
Deleting an IP Pool 2-9
Updating an IP Pool 2-9

Listing, Creating, Deleting, and Modifying COS Node Clusters 2-10
Listing the COS Node Clusters 2-10
Listing One COS Node Cluster 2-10
Creating a COS Node Cluster 2-11
Deleting a COS Node Cluster 2-11
Updating a COS Node Cluster 2-11

Listing, Creating, Deleting, and Modifying COS Nodes 2-12
Listing the COS Nodes 2-12
Listing One COS Node 2-13
Creating a COS Node 2-13
Deleting a COS Node 2-14
Updating a COS Node 2-14

Listing, Creating, Deleting, and Modifying Auth Profiles 2-15
Listing the Auth Profiles 2-15
Listing One Auth Profile 2-16
Creating an Auth Profile 2-16
Deleting an Auth Profile 2-17
Updating an Auth Profile 2-17

Listing and Modifying Service Instances 2-18
Listing the Service Instances 2-18
Listing One Service Instance 2-19
Updating a Service Instance 2-20

Listing, Creating, Deleting, and Modifying Service Endpoints 2-21
Cisco Cloud Object Storage Release 2.1.1 API Guide

Contents

CHAPTER 2
Listing the Service Endpoints  2-21
Listing One Service Endpoint  2-22
Creating a Service Endpoint  2-22
Deleting a Service Endpoint  2-23
Updating a Service Endpoint  2-23
Listing, Creating, Deleting, and Modifying Asset Workflow Templates  2-24
Listing the Asset Workflow Templates  2-25
Listing One Asset Workflow Templates  2-25
Creating a Workflow Templates  2-25
Deleting a Workflow Templates  2-26
Updating a Workflow Templates  2-27
Viewing COS Node Status  2-28
Viewing the Status of COS Nodes  2-28
Viewing the Status of a specific COS Node  2-29
Viewing Events for Nodes  2-29
Viewing the Events for Nodes in a Region  2-29
Viewing the Events for Nodes in a Service Instance  2-30
Viewing Application Status for the COS Instance  2-31

CHAPTER 3
Swauth API  3-1
Listing Accounts  3-1
Retrieving Account Details  3-2
Creating an Account  3-3
Deleting an Account  3-4
Creating or Updating a User  3-4
Retrieving User Details  3-5
Deleting a User  3-6
Creating or Updating Account Service Endpoints  3-7
Getting an Authentication Token  3-8

CHAPTER 4
Swift API  4-1
Listing Containers  4-1
Listing Objects  4-2
Creating a Container  4-3
Deleting a Container  4-4
Retrieving an Object  4-4
Creating or Updating An Object  4-7
Preface

This preface describes who should read the Cisco Cloud Object Storage Release 2.1.1 API Guide, how it is organized, and its document conventions. It contains the following sections:

- Audience
- Document Organization
- Document Conventions
- Related Publications
- Obtaining Documentation and Submitting a Service Request

Audience

This application program interface (API) guide is written for the knowledgeable application programmer who understands the basic architecture of the Cisco Cloud Object Storage (COS) product and Java servlets. The user should be fluent in the Java programming language and have prior practical experience developing content networking solutions. This guide is not intended to direct the user in how to program in the Java language and limits itself to describing how related COS software servlets are used.

Document Organization

This document contains the following chapters and appendices:

<table>
<thead>
<tr>
<th>Chapters or Appendices</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 1, “Introduction to COS and COS APIs”</td>
<td>Introduces the COS software APIs</td>
</tr>
<tr>
<td>Chapter 2, “Service Manager API”</td>
<td>Describes the subset of the MOS APIs that are implemented for the COS.</td>
</tr>
<tr>
<td>Chapter 3, “Swauth API”</td>
<td>Describes the subset of the OpenStack Swauth API that is implemented for the COS authentication service.</td>
</tr>
<tr>
<td>Chapter 4, “Swift API”</td>
<td>Describes the subset of the OpenStack Swift API that is implemented for COS.</td>
</tr>
<tr>
<td>Appendix A, “Example API Calls”</td>
<td>Provides examples for making Service Manager, Swauth, and Swift API calls using curl.</td>
</tr>
</tbody>
</table>
Document Conventions

This document uses the following conventions:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Indication</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bold</strong> font</td>
<td>Commands and keywords and user-entered text appear in <strong>bold</strong> font.</td>
</tr>
<tr>
<td><strong>italic</strong> font</td>
<td>Document titles, new or emphasized terms, and arguments for which you supply values are in <strong>italic</strong> font.</td>
</tr>
<tr>
<td>[ ]</td>
<td>Elements in square brackets are optional.</td>
</tr>
<tr>
<td>{x</td>
<td>y</td>
</tr>
<tr>
<td>[x</td>
<td>y</td>
</tr>
<tr>
<td>string</td>
<td>A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.</td>
</tr>
<tr>
<td><strong>courier</strong> font</td>
<td>Terminal sessions and information the system displays appear in <strong>courier</strong> font.</td>
</tr>
<tr>
<td>&lt; &gt;</td>
<td>Nonprinting characters such as passwords are in angle brackets.</td>
</tr>
<tr>
<td>[ ]</td>
<td>Default responses to system prompts are in square brackets.</td>
</tr>
<tr>
<td>!, #</td>
<td>An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.</td>
</tr>
</tbody>
</table>

### Note

Means reader take note. Notes contain helpful suggestions or references to material not covered in the manual.

### Tip

Means the following information will help you solve a problem. The tips information might not be troubleshooting or even an action, but could be useful information, similar to a Timesaver.

### Caution

Means reader be careful. In this situation, you might perform an action that could result in equipment damage or loss of data.

### Timesaver

Means the described action saves time. You can save time by performing the action described in the paragraph.

### Warning

**IMPORTANT SAFETY INSTRUCTIONS**

This warning symbol means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Use the statement number provided at the end of
each warning to locate its translation in the translated safety warnings that accompanied this device.

SAVE THESE INSTRUCTIONS

---

Warning

Statements using this symbol are provided for additional information and to comply with regulatory and customer requirements.

Related Publications

Refer to the following documents for additional information about COS 2.1.1:

- Release Notes for Cisco Cloud Object Storage Release 2.1.1
- Cisco Content Delivery Engine 205/220/250/420/460/470 Hardware Installation Guide
- Cisco Media Origination System Release 2.2 User Guide
- Open Source Used in Cisco Cloud Object Storage Release 2.1.1

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, using the Cisco Bug Search Tool (BST), submitting a service request, and gathering additional information, see What’s New in Cisco Product Documentation at: http://www.cisco.com/c/en/us/td/docs/general/whatsnew/whatsnew.html.

Subscribe to What’s New in Cisco Product Documentation, which lists all new and revised Cisco technical documentation, as an RSS feed and deliver content directly to your desktop using a reader application. The RSS feeds are a free service.
Introduction to COS and COS APIs

Product Description

The Cisco Cloud Object Storage (COS) provides distributed, resilient, high-performance storage and retrieval of binary large object (blob) data. The primary interface for managing COS content is the OpenStack Swift API, with enhancements that improve the quality of service when accessing large media objects.

Storage is distributed across a cluster of hardware systems, or nodes. The storage cluster is resilient against hard drive failure within a node and against node failure within the cluster. Nodes may be added to or removed from the cluster as needed to provide for changes in cluster capacity. To administer the cluster, COS includes an HTTP-based cluster-management API.

COS also includes an authentication and authorization service that implements the OpenStack Swauth API.

COS and MOS

COS is designed to integrate transparently with the Cisco Media Origination System (MOS), which is designed for highly optimized ingest and storage. MOS uses a hierarchical storage design that supports huge content libraries while simplifying content storage management. Its distributed architecture can separate ingest and storage from streaming, allowing each function to be scaled independently as needed to dynamically increase network ingest and storage resources.

Components

COS has a number of subsystems:

- **Networks:** Interfaces are grouped into distinct networks to isolate management functions from high-volume data traffic. The client applications use Swauth API to interact with the COS authentication and authorization services, and the Swift API to interact with the COS object storage services.

- **Clusters and Nodes:** COS services are provided by a cluster of nodes, with both the cluster and the individual nodes as distinctly manageable components.

- **Object Metadata Store:** The metadata for the cluster is stored in a high-performance distributed NoSQL database hosted on the COS nodes in a cluster.
- **Platform and Applications Manager (PAM):** COS components are managed using services running on the Platform and Applications Management system (PAM).
- **Hardware Platforms:** COS software is deployed on Cisco Content Delivery Engine (CDE) platforms.

### API Features

- **Overview, page 1-2**
- **Service Manager API, page 1-3**
- **Swauth API, page 1-3**
- **Swift Object Store API, page 1-4**

### Overview

The table below provides an overview of the COS APIs

<table>
<thead>
<tr>
<th>Feature Set</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Manager API</td>
<td>• A subset of the Cisco Media Origination System (MOS) APIs</td>
</tr>
<tr>
<td></td>
<td>• Used to provision and configure a COS cluster and COS cluster nodes</td>
</tr>
<tr>
<td></td>
<td>• Uses the FQDN of the Service Manager and HTTPS over port 8043</td>
</tr>
<tr>
<td>Swauth API</td>
<td>• Simple Auth Service API for authentication of Swift operations</td>
</tr>
<tr>
<td></td>
<td>• Based on Swauth Open-Source Middleware API</td>
</tr>
<tr>
<td></td>
<td>• Used to manage accounts, users, and account service endpoints</td>
</tr>
<tr>
<td></td>
<td>• Uses the Authentication FQDN of the COS cluster and HTTP over port 80</td>
</tr>
<tr>
<td>Swift Object Store API</td>
<td>• An implementation of a subset of the continually evolving OpenStack Swift API</td>
</tr>
<tr>
<td></td>
<td>• Command executions are authenticated using auth tokens provided by Swauth service</td>
</tr>
<tr>
<td></td>
<td>• Used to create and manage containers and objects for persistent storage in a COS cluster</td>
</tr>
<tr>
<td></td>
<td>• Uses the Storage FQDN of the COS cluster and HTTP over port 80</td>
</tr>
</tbody>
</table>
Chapter 1      Introduction to COS and COS APIs

API Features

Note
The COS cluster is assigned an Authentication FQDN (used with the Swauth API) and a Storage FQDN (used with the Swift API). Currently the Authentication FQDN and the Storage FQDN must be the same, for example, auth01.cos.acme.com.

Service Manager API

The COS Service Manager API is a subset of the MOS APIs. The Service Manager API provides the following functions:

- Listing the region
- Listing, creating, deleting, and modifying zones
- Listing, creating, deleting, and modifying IP pools
- Listing, creating, deleting, and modifying a COS cluster
- Adding a COS node to a COS cluster
- Listing, editing, and enabling a COS instance
- Listing, creating, deleting, and modifying a COS service endpoint
- Listing, creating, deleting, and modifying an asset workflow template
- Listing and modifying COS nodes
- Configuring a COS node service interface
- Viewing the COS node status
- Viewing the COS node management events

For a detailed description of these functions, see “Service Manager API”.

Swauth API

COS includes a basic authentication service that can be used when COS is not installed along with other OpenStack services such as the Keystone Identity service. The API for the COS authentication service is derived from the OpenStack Swauth middleware component API. The authentication service API provides the following functions for managing accounts, users, and service endpoints:

- Listing Accounts
- Retrieving Account Details
- Creating an Account
- Deleting an Account
- Creating or Updating a User
- Retrieving User Details
- Deleting a User
- Creating or Updating Account Service Endpoints
- Getting an Authentication Token

For a detailed description of these functions, see “Swauth API”.
The COS cluster is assigned an Authentication FQDN (used with the Swauth API) and a Storage FQDN (used with the Swift API). Currently the Authentication FQDN and the Storage FQDN must be the same, for example, auth01.cos.acme.com.

**Swift Object Store API**

The COS object storage API is based on the OpenStack Swift API. It is implemented as a set of Representational State Transfer (ReSTful) web services. All account, container, and object operations can be performed with standard HTTP calls. The requests are directed to the host and URL described in the X-Storage-Url HTTP header that is part of the response to a successful request for an authentication token.

The COS object storage API defines restrictions on HTTP requests. These restrictions, borrowed from the Swift API, are listed in the table below.

**Table 1-2 COS API Restrictions**

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum # of HTTP Headers per request</td>
<td>90</td>
</tr>
<tr>
<td>Maximum length of all HTTP Headers</td>
<td>4096 bytes</td>
</tr>
<tr>
<td>Maximum length per HTTP request line</td>
<td>8192 bytes</td>
</tr>
<tr>
<td>Maximum length of container name</td>
<td>256 bytes</td>
</tr>
<tr>
<td>Maximum length of object name</td>
<td>1024 bytes</td>
</tr>
</tbody>
</table>

Also, the container and object names must be UTF-8 encoded and then URL-encoded before inclusion in the HTTP request line.

All the length restrictions are enforced against the URL-encoded request line.

The COS object store API provides the following functions, some of which provide extended functionality beyond the standard SWIFT API defined by OpenStack:

- Listing Containers
- Listing Objects
- Creating a Container
- Deleting a Container
- Retrieving an Object
- Creating or Updating An Object
- Deleting an Object
- Creating or Updating Container Metadata
- Retrieving Container Metadata
- Deleting Container Metadata
- Retrieving Object Metadata

For a detailed description of these functions, see “Swift API”. 
Restrictions and Limitations

- The OpenStack Swift and Swauth APIs continue to evolve. COS does not currently implement all the Swift or Swauth API functions. For a list of supported functions, see Swift Object Store API and Swauth API in this chapter.
- Secure Sockets Layer (SSL) or other means for providing session security and encryption are not supported with the Swift and Swauth APIs.
- The Service Manager API support access using HTTPS over port 8043.
- See the COS 2.1.1 Release Notes for open caveats and known issues related to this release of the COS software.
Service Manager API

This chapter describes the subset of the Cisco MOS API that is implemented for COS. The Service Manager API uses the FQDN of the Service Manager and HTTPS over port 8043.

Listing the Region

In COS 2.1.1, only one Region is supported. The object is named ‘region-0’ and can only be read. To retrieve information about the region, use the following HTTP GET request:

**GET https://SM_FQDN:8043/v2/regions**

This command returns a list of region objects:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
{}
```

Listing DNS Servers

The DNSServer object defines the Name server for COS. This is used to insert the COS A Records and for load balancing components within COS. Depending on the choice made at installation, the DNSServer can be supported on the COS PAM or provided by an external DNS server.

When PAM acts as the DNSServer, all PAM nodes (high availability) act as redundant DNS Servers. This object is setup at install time and cannot be changed.

Listing all DNS Servers

To retrieve a list of all DNS Servers, use the following HTTP GET request:

**GET https://SM_FQDN:8043/v2/regions/region_name/dnsservers**
This command returns a list of external DNS Server objects that have been configured:

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

```json
{
  "id": "smregion_0.smdnsserver.extdns-1",
  "name": "extdns-1",
  "type": "dnsservers",
  "externalId": "/v2/regions/region-0/dnsservers/extdns-1",
  "transactionId": "d29db01f-49bf-49bf-9288-0e4fe6f51f39",
  "properties": {
    "description": "dns server",
    "ipAddr": "10.93.232.70",
    "domain": "mos.npi.cds.cisco.com",
    "authType": "tsig",
    "tsigKey": "k1fj1MVWV1S610K0vVl7w==&quot;",
    "tsigAlgorithm": "hmac-md5"
  }
}
```

**Listing One DNS Server**

GET https://SM_FQDN:8043/v2/regions/region_name/dnsservers/dnsservername

**Listing and Modifying NTP Servers**

There is one NTP Server Object per region. Each object may have one or more NTP Server instances defined.

**Listing all NTP Servers**

To retrieve a list of all NTP server objects, use the following HTTP GET request:

GET https://SM_FQDN:8043/v2/regions/region_name/ntpservers

This command returns a list of NTP Server objects:

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

```json
{
  "id": "smregion_0.smntpserver.ntp-1",
  "name": "ntp-1",
  "type": "ntpservers",
  "externalId": "/v2/regions/region-0/ntpservers/ntp-1",
  "transactionId": "19a0b7a1-de88-45f7-9593-8d052aa3913b",
  "properties": {
    "description": "ntp server list",
    "servers": [
      "10.81.254.202",
      "10.81.254.131"
    ]
  }
}
```

**Listing one NTP Server**

GET https://SM_FQDN:8043/v2/regions/region_name/ntpservers/ntpservername
### Updating an NTP Server

**PUT** https://SM_FQDN:8043/v2/regions/region_name/ntpservers/ntpservername

**Example:**
- **Request curl:**
  
  curl -k -H "Content-Type: application/json" -X PUT --data @body.json https://10.10.150.22:8043/v2/regions/region-0/ntpservers/ntp-1

  - @body.json file format:

    ```json
    {
      "properties": {
        "description": "ntp server list",
        "servers": ["10.22.171.18"]
      }
    }
    ```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>servers</td>
<td>Array of strings</td>
<td>Yes</td>
<td>Comma separated list of the IP addresses or hostnames of the NTP servers</td>
<td></td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Listing DNS Forwarders

The DNSForwarder object defines the Query server for COS. This is used to resolve the DNS names by COS components. This object is setup at PAM install time and can be updated afterwards.

#### Listing DNS Forwarders

To retrieve a list of all DNS Forwarders, use the following HTTP GET request:

**GET** https://SM_FQDN:8043/v2/regions/region_name/dnsforwarders

This command returns a list of DNS Forwarder objects:

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

```json
{
  "id": "smregion_0.smdnsforwarder.dnsFwd-1",
  "name": "dnsFwd-1",
  "type": "dnsforwarders",
  "externalId": "/v2/regions/region-0/dnsforwarders/dnsFwd-1",
  "transactionId": "bcff4a4c-00ac-4519-8beb-b3d7b5c7b3e4",
  "properties": {
    "description": "dns forwarder",
    "ipAddr": "10.93.232.70",
    "domain": "mos.npi.cds.cisco.com"
  }
}
```
Listing One DNS Forwarder

GET https://SM_FQDN:8043/v2/regions/region_name/dnsforwarders/dnsforwardername

Listing, Creating, Deleting, and Modifying Zones

The zone is in the scope of a region. The regionRef parameter is the ‘id’ of the region object in which the zone exists.

Listing All Zones

To retrieve a list all of the zones, use the following HTTP GET request:

GET https://SM_FQDN:8043/v2/zones

This command returns a list of zone objects:

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

[{
   "id": "smtenant_system.smzone.zone-1",
   "name": "zone-1",
   "type": "zones",
   "externalId": "/v2/zones/zone-1",
   "properties": {
       "description": "A Zone sample in region-0",
       "regionRef": "smtenant_system.smregion.region-0"
   }
}]

Listing One Zone

GET https://SM_FQDN:8043/v2/zones/zoneName

Creating a Zone

POST https://SM_FQDN:8043/v2/zones/zoneName

Example:

- Request curl:
  curl -k -H "Content-Type: application/json" -X POST --data @body.json https://10.10.150.22:8043/v2/zones/east

- @body.json file format:

  {
   "properties": {
       "description": "A Zone sample in region-0",
       "regionRef": "smtenant_system.smregion.region-0"
   }
  }
Deleting a Zone

DELETE https://SM_FQDN:8043/v2/zones/zoneName

Updating a Zone

PUT https://SM_FQDN:8043/v2/zones/zoneName

Example:
- Request curl:
  ```
curl -k -H "Content-Type: application/json" -X PUT --data @body.json https://10.150.22:8043/v2/zones/east
  ```
- @body.json file format:
  ```
  
  "properties": {
    "description": "A Zone in the sm region",
    "regionRef": "smtenant_system.smregion.region-0"
  }
  ```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>regionRef</td>
<td>String</td>
<td>Yes</td>
<td>.<em>..smregion..</em></td>
<td>id property of a region document, of which the zone belongs to</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Listing, Creating, Deleting, and Modifying Networks

Before creating IP pools, you must configure networks.

Listing the Networks

To retrieve a list all of the Network objects, use the following HTTP GET request:

GET https://SM_FQDN:8043/v2/networks

This command returns a list of Network objects:
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

```json
{
  "id": "smtenant_system.smnetwork.network-a",
  "name": "network-a",
  "type": "networks",
  "externalId": "/v2/networks/network-a",
  "properties": {
    "netName": "red",
    "zoneRef": "smtenant_system.smzone.zone-1",
    "fqdn": "internal.mos.com"
  }
}
```

**Listing One Network**

GET `https://SM_FQDN:8043/v2/networks/network_name`

**Creating a Network**

POST `https://SM_FQDN:8043/v2/networks/network_name`

**Example:**

- **Request curl:**
  ```
  curl -k -H "Content-Type: application/json" -X POST --data @body.json
  ```

- **@body.json file format:**
  ```
  {
    "properties": {
      "netName": "south",
      "zoneRef": "smtenant_system.smzone.zone-1"
    }
  }
  ```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>netName</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>If a name is reused in a zone, then the ipPoolRef must be identical.</td>
</tr>
<tr>
<td>zoneRef</td>
<td>String</td>
<td>Yes</td>
<td>.<em>./.smzone.</em></td>
<td>Zone to which the network belongs to.</td>
</tr>
<tr>
<td>fqdn</td>
<td>String</td>
<td>No</td>
<td>^[a-z0-9]+(-[a-z0-9]+)*.[a-z][2,6]$</td>
<td>For example: mosint.acme.com</td>
</tr>
</tbody>
</table>
Deleting a Network

DELETE https://SM_FQDN:8043/v2/networks/network_name

Updating a Network

PUT https://SM_FQDN:8043/v2/networks/network_name

Example:
• Request curl:

```
curl -k -H "Content-Type: application/json" -X PUT --data @body.json
```

• @body.json file format:

```
{
    "properties": {
        "netName": "south",
        "zoneRef": "smtenant_system.smzone.zone-1"
    }
}
```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>netName</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>If a name is reused in a zone, then the ipPoolRef must be identical.</td>
</tr>
<tr>
<td>zoneRef</td>
<td>String</td>
<td>Yes</td>
<td>.<em>.smzone..</em></td>
<td>Zone to which the network belongs to.</td>
</tr>
<tr>
<td>fqdn</td>
<td>String</td>
<td>No</td>
<td>^[a-z0-9]+([-.]{1}[a-z0-9]+)*.[a-z]{2,6}$</td>
<td>For example: mosint.acme.com</td>
</tr>
</tbody>
</table>

Listing, Creating, Deleting, and Modifying IP Pools

Before installing COS nodes, you must configure IP pools.

Listing the IP Pools

To retrieve a list all of the IP pools, use the following HTTP GET request:

GET https://SM_FQDN:8043/v2/ippools

This command returns a list of IP Pool objects:

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Listing One IP Pool

GET https://SM_FQDN:8043/v2/ippools/IPPoolName

Creating an IP Pool

POST https://SM_FQDN:8043/v2/ippools/IPPoolName

Example:
• Request curl:
  curl -k -H "Content-Type: application/json" -X POST --data @body.json https://10.10.150.22:8043/v2/ippools/ippool-2

• @body.json file format:

```json
{
  "properties": {
    "addrType": "ipv4",
    "networkRef": "smttenant_system.smnetwork.network-b",
    "pool": [
      {
        "rangeStart": "10.22.22.1",
        "rangeEnd": "10.22.22.25",
        "netmask": "255.255.255.224",
        "gw": "10.22.22.30"
      }
    ]
  }
}
```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>addrType</td>
<td>String</td>
<td>Yes</td>
<td>“ipv4”</td>
<td>Currently the only valid choice is ipv4.</td>
</tr>
</tbody>
</table>
### Deleting an IP Pool

**DELETE** https://SM_FQDN:8043/v2/ippools/IPPoolName

### Updating an IP Pool

**PUT** https://SM_FQDN:8043/v2/ippools/IPPoolName

**Example:**

- **Request curl:**
  ```
curl -k -H "Content-Type: application/json" -X PUT --data @body.json https://10.10.150.22:8043/v2/ippools/ippool-2
  ```

- **@body.json file format:**
  ```json
  {
    "properties": {
      "addrType": "ipv4",
      "networkRef": "smtenant_system.smnetwork.network-b",
      "pool": [
        {
          "rangeStart": "10.22.22.1",
          "rangeEnd": "10.22.22.25",
          "netmask": "255.255.255.224",
          "gw": "10.22.22.30"
        }
      ]
    }
  }
  ```
Listing, Creating, Deleting, and Modifying COS Node Clusters

Listing the COS Node Clusters

To retrieve a list of all COS node clusters, use the following HTTP GET request:

GET https://SM_FQDN:8043/v2/regions/region_name/cosnodeclusters

This command returns a list of COS node cluster objects:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

[{
  "id": "smregion_0.smcosnodecluster.cluster.ca.01",
  "name": "cluster.ca.01",
  "type": "cosnodeclusters",
  "externalId": "/v2/regions/region-0/cosnodeclusters/cluster.ca.01",
  "transactionId": "bf0e75ce-bda1-46f2-9c50-3a66ae34af33",
  "properties": {
    "description": "A sample cos node cluster",
    "authFqdn": "auth01.cos1.acme.com",
    "storageFqdn": "auth01.cos1.acme.com",
    "dnsServersRef": "",
    "regionRef": "smttenant_system.smregion.region-0"
  }
}]
```

Listing One COS Node Cluster

GET https://SM_FQDN:8043/v2/regions/region_name/cosnodeclusters/cluster_name
Creating a COS Node Cluster

**POST** https://SM_FQDN:8043/v2/regions/region_name/cosnodeclusters/cluster_name

**Example:**
- **Request curl:**
  ```
  curl -k -H "Content-Type: application/json" -X POST --data @body.json https://10.10.150.22:8043/v2/regions/region-0/cosnodeclusters/cluster.az.01
  ```
- **@body.json file format:**

  ```json
  {
    "properties": {
      "description": "A cos node cluster in arizona",
      "authFqdn": "auth01.cos1.acme.com",
      "storageFqdn": "auth01.cos1.acme.com"
    }
  }
  ```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>authFqdn</td>
<td>String</td>
<td>No</td>
<td>^[a-z0-9]+([-./]{1}[a-z0-9]+)*([-./]{1}[a-z0-9]+){2,6}$</td>
<td>^$</td>
</tr>
<tr>
<td>storageFqdn</td>
<td>String</td>
<td>No</td>
<td>^[a-z0-9]+([-./]{1}[a-z0-9]+)*([-./]{1}[a-z0-9]+){2,6}$</td>
<td>^$</td>
</tr>
</tbody>
</table>

Deleting a COS Node Cluster

**DELETE** https://SM_FQDN:8043/v2/regions/region_name/cosnodeclusters/cluster_name

Updating a COS Node Cluster

**PUT** https://SM_FQDN:8043/v2/regions/region_name/cosnodeclusters/cluster_name

**Example:**
- **Request curl:**
  ```
  curl -k -H "Content-Type: application/json" -X PUT --data @body.json https://10.10.150.22:8043/v2/regions/region-0/cosnodeclusters/cluster.az.01
  ```
- **@body.json file format:**
Listing, Creating, Deleting, and Modifying COS Nodes

Listing the COS Nodes

To retrieve a list of all COS nodes, use the following HTTP GET request:

GET https://SM_FQDN:8043/v2/cosnodes

This command returns a list of COS node objects:

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
    "id": "smtenant_system.smcosnode.173926472",
    "name": "173926472",
    "type": "cosnodes",
    "externalId": "/v2/cosnodes/173926472",
    "transactionId": "a3ce7f68-4e3c-457c-b24a-647bb185f7c9",
    "properties": {
        "dataInterfaces": [
            {
                "name": "eth2",
                "ipPoolRef": "smtenant_system.smippool.2",
                "enabled": true
            }
        ]
    }
}
"zoneRef": "smtenant_system.smzone.zone-1",
"model": "",
"description": "A COS Node appliance",
"cosNodeClusterRef": "smregion_0.smcosnodecluster.cluster.ca.01",
"mgmtAddress": "10.93.232.72",
"adminState": "inservice"
}
}

Listing One COS Node

GET https://SM_FQDN:8043/v2/cosnodes/node_name

Creating a COS Node

POST https://SM_FQDN:8043/v2/cosnodes/node_name

Example:

- Request curl:

```
curl -k -H "Content-Type: application/json" -X POST --data @body.json https://10.10.150.22:8043/v2/cosnodes/173926473
```
- `@body.json` file format:

```json
{
  "properties": {
    "dataInterfaces": [
      {
        "name": "eth2",
        "ipPoolRef": "smtenant_system.smippool.emer1",
        "enabled": true
      }
    ],
    "zoneRef": "smtenant_system.smzone.zone-1",
    "model": "",
    "description": "A COS Node appliance in zone-1",
    "cosNodeClusterRef": "smregion_0.smcosnodecluster.cluster.ca.01",
    "mgmtAddress": "10.22.22.79",
    "adminState": "inservice"
  }
}
```

Note

When creating a COS node, the cosNodeClusterRef value should be set to the id value that is returned for the COS Node Cluster using GET https://SM_FQDN:8043/v2/regions/region_name/cosnodeclusters/cluster_name

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>zoneRef</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>ID property of a zone document, to which the node belongs</td>
</tr>
<tr>
<td>model</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>For example: CDE460-4R1</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Deleting a COS Node

DELETE `https://SM_FQDN:8043/v2/cosnodes/node_name`

### Updating a COS Node

PUT `https://SM_FQDN:8043/v2/cosnodes/node_name`

**Example:**
- **Request curl:**
  ```bash
curl -k -H "Content-Type: application/json" -X PUT --data @body.json
https://10.10.150.22:8043/v2/cosnodes/173926473
```
- **@body.json file format:**
  ```json
  {
    "properties": {
      "dataInterfaces": [
        {
          "name": "eth2",
          "ipPoolRef": "smtenant_system.smippool.emer1",
          "enabled": true
        }
      ],
      "zoneRef": "smtenant_system.smzone.zone-1",
      "model": "",
      "description": "A COS Node appliance in zone-1",
      "cosNodeClusterRef": "smregion_0.smcosnodecluster.cluster.ca.01",
      "mgmtAddress": "10.22.22.79",
      "adminState": "inservice"
    }
  }
  ```
Listing, Creating, Deleting, and Modifying Auth Profiles

Listing the Auth Profiles

To retrieve a list of all Auth profiles, use the following HTTP GET request:

**GET https://SM_FQDN:8043/v2/authprofiles**

This command returns a list of Auth profile objects:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
{
   "id": "smtenant_0.smauthprofile.auth12345",
   "name": "auth12345",
   "type": "authprofiles",
   "externalId": "/v2/authprofiles/auth12345",
   "transactionId": "24de4734-91be-4b22-a444-f33e9cd2a6d3",
   "properties": {
      "type": "tmpauth",
```

---

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>zoneRef</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>id property of a zone document, of which the node belongs to. Manually updated to point to zone object</td>
</tr>
<tr>
<td>model</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>Eg: CDE460-4R1</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cosNodeClusterRef</td>
<td>String</td>
<td>No</td>
<td></td>
<td>id property of a document, of which the node belongs to</td>
</tr>
<tr>
<td>mgmtAddress</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>IP address of the management server</td>
</tr>
<tr>
<td>adminState</td>
<td>Enum</td>
<td>Yes</td>
<td></td>
<td>adminState can be one of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• inservice</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• maintenance</td>
</tr>
<tr>
<td>dataInterfaces</td>
<td>Array</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>Unique name within node. For example: eth1</td>
</tr>
<tr>
<td>ipPoolRef</td>
<td>String</td>
<td>No</td>
<td>^***.smippool..*</td>
<td>^$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>For example: smtenant_system.smi ppool.emer1</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Yes</td>
<td></td>
<td>Enter true or false</td>
</tr>
</tbody>
</table>

---

Listing, Creating, Deleting, and Modifying Auth Profiles
Listing, Creating, Deleting, and Modifying Auth Profiles

Listing One Auth Profile

GET https://SM_FQDN:8043/v2/authprofiles/authprofile_name

Creating an Auth Profile

POST https://SM_FQDN:8043/v2/authprofiles/authprofile_name

Example:

- Request curl:

  curl -k -H "Content-Type: application/json" -X POST --data @body.json

- @body.json file format:

  
  ```
  {
    "properties": { 
      "type": "swauth",
      "description": "profile for cos",
      "userid": "jbrown:user1",
      "accesskey": "tk7ec41e82175a42918c13fd4d172c9399",
      "server": { 
        "authServerUrl": "http://auth01.cos1.acme.com"},
      "tokenRefreshIntervalSec": "3600"
    }
  }
  ```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>enum</td>
<td>Yes</td>
<td></td>
<td>type can be one of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• tmpauth</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• keystone</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• swauth</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>userid</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>accesskey</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tokenRefreshIntervalSec</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>Token refresh interval in seconds</td>
</tr>
<tr>
<td>server</td>
<td>object</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Deleting an Auth Profile

**DELETE** https://SM_FQDN:8043/v2/authprofiles/authprofile_name

Updating an Auth Profile

**PUT** https://SM_FQDN:8043/v2/authprofiles/authprofile_name

Example:

- Request curl:

  ```
curl -k -H "Content-Type: application/json" -X PUT --data @body.json https://10.10.150.22:8043/v2/authprofiles/auth-2
  ```

- `@body.json` file format:

  ```
  {
    "properties": {
      "type": "swauth",
      "description": "profile for cos",
      "userid": "jbrown:user1",
      "accesskey": "tk7ec41e82175a42918c13fd4d172c9399",
      "server": {
        "authServerUrl": "http://auth01.cos1.acme.com",
        "tokenRefreshIntervalSec": "3600"
      }
    }
  }
  ```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server Properties</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>authServerUrl</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| type                   | enum     | Yes      |               | type can be one of the following:  
  - tmpauth  
  - keystone  
  - swauth    |

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>userid</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>accesskey</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tokenRefreshIntervalSec</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>Token refresh interval in seconds</td>
</tr>
<tr>
<td>server</td>
<td>object</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authServerUrl</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Listing and Modifying Service Instances

The ServiceInstance object represents a Service Instance of COS. A Service Instance is produced by using the Nodes for capture and playback endpoints. The endpoints are selected using the AssetWorkflowTemplate. There can be one or more AssetWorkflowTemplates in a Service Instance. The Service Instance uses ImageManifest to specify which images (and hence Nodes) are to be selected for software instantiation and use.

The Service Instance can be stopped or started using the state parameter. The templateRef parameter specifies the Service Template that will control the Service Instance behavior. The title is a customizable name of the Service Instance. This name shows up on the GUI as the instance name.

In COS 2.1.1, the Service Instance can only be Read and Updated. A set of five live and five VoD Service Instances in inactive state come preinstalled. These can be customized and used (activated).

Listing the Service Instances

To retrieve a list of all Service Instances, use the following HTTP GET request:

GET https://SM_FQDN:8043/v2/serviceinstances

This command returns a list of service instance objects:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

[
  {
    "id": "smtenant_0.smserviceinstance.cos-0-1",
    "name": "cos-0-1",
    "type": "serviceinstances",
    "externalId": "/v2/serviceinstances/cos-0-1",
    "transactionId": "a885b9fd-9199-4834-81b3-878603a7f882",
    "properties": {
      "state": "active",
      "title": "An unused COS Service",
      "regionsRef": [
        "smtenant_system.smregion.region-0"
      ],
      "imageManifestRef": "smtenant_0.smimagemanifest.image-manifest-1",
      "description": "COS service",
      "templateRef": "smbase.sm servicetemplate.cos_0.v1",
      "customConfigs": []
    }
  },
  {
    "id": "smtenant_0.smserviceinstance.ums-0-2",
    "name": "ums-0-2",
    "type": "serviceinstances",
    "externalId": "/v2/serviceinstances/ums-0-2",
    "properties": {
      "state": "inactive",
      "title": "An unused UMS Service",
      "regionsRef": [],
      "description": "A sample UMS instance",
      "templateRef": "smbase.sm servicetemplate.ums_0.v1",
      "customConfigs": []
    }
  },
  {
    "id": "smtenant_0.smserviceinstance.ums-0-3",
    "name": "ums-0-3",
    "type": "serviceinstances",
  }
]```
Listing and Modifying Service Instances

- **externalId**: "/v2/serviceinstances/ums-0-3",
  - **properties**: {
    - **state**: "inactive",
    - **title**: "An unused UMS Service",
    - **regionsRef**: [],
    - **description**: "A sample UMS instance",
    - **templateRef**: "smbase.sm servicetemplate.ums_0.v1",
    - **customConfigs**: []
  }
},
- **id**: "smtenant_0.sm serviceinstance.ums-0-4",
  - **name**: "ums-0-4",
  - **type**: "serviceinstances",
  - **externalId**: "/v2/serviceinstances/ums-0-4",
  - **properties**: {
    - **state**: "inactive",
    - **title**: "An unused UMS Service",
    - **regionsRef**: [],
    - **description**: "A sample UMS instance",
    - **templateRef**: "smbase.sm servicetemplate.ums_0.v1",
    - **customConfigs**: []
  }
},
- **id**: "smtenant_0.sm serviceinstance.ums-0-5",
  - **name**: "ums-0-5",
  - **type**: "serviceinstances",
  - **externalId**: "/v2/serviceinstances/ums-0-5",
  - **properties**: {
    - **state**: "inactive",
    - **title**: "An unused UMS Service",
    - **regionsRef**: [],
    - **description**: "A sample UMS instance",
    - **templateRef**: "smbase.sm servicetemplate.ums_0.v1",
    - **customConfigs**: []
  }
},
- **id**: "smtenant_0.sm serviceinstance.ums-0-1",
  - **name**: "ums-0-1",
  - **type**: "serviceinstances",
  - **externalId**: "/v2/serviceinstances/ums-0-1",
  - **transactionId**: "d61bdc5e-483c-4854-b7dd-80d05deeda5f",
  - **properties**: {
    - **state**: "active",
    - **title**: "SEVT UMS Service",
    - **regionsRef**: [],
    - **description**: "SEVT UMS instance",
    - **templateRef**: "smbase.sm servicetemplate.ums_0.v1",
    - **customConfigs**: [],
    - **imageManifestRef**: "smtenant_0.sm imagemanifest.image-manifest-1"
  }
}

### Listing One Service Instance

**GET** https://SM_FQDN:8043/v2/serviceinstances/serviceinstance_name

The following is an example output for listing the COS Service Instance:

```json
[
  {
    "id": "smtenant_0.sm serviceinstance.cos-0-1",
```
Updating a Service Instance

PUT https://SM_FQDN:8043/v2/serviceinstances/serviceinstance_name

Example:

- Request curl:
  
curl -k \-H "Content-Type: application/json" \-X PUT --data @body.json https://10.10.150.22:8043/v2/serviceinstances/cos-0-1

- @body.json file format:

```json
{
  "properties": {
    "state": "active",
    "title": "COS Service 2",
    "regionsRef": [
      "smtenant_system.smregion.region-0",
    ],
    "imageManifestRef": "smtenant_0.smimagemanifest.image-manifest-1",
    "description": "COS service",
    "templateRef": "smbase.smservietemplate.cos_0.v1",
    "customConfigs": []
  }
}
```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>enum</td>
<td>Yes</td>
<td>state can be one of the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- inactive</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- init</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- validating</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- active</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- deleting</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- deleted</td>
<td></td>
</tr>
</tbody>
</table>
Listing, Creating, Deleting, and Modifying Service Endpoints

A service endpoint is a logical configuration reference point to defined resources and policies for the capture, ingest, and storage of content in COS. One Service Instance can have one or more service endpoints (also referred to as capture endpoints) objects.

Listing the Service Endpoints

To retrieve a list of all service endpoints, use the following HTTP GET request:

GET https://SM_FQDN:8043/v2/serviceinstances/serviceinstance_name/captureendpoints

This command returns a list of service endpoint (capture endpoint) objects:

---

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>Customer provided short displayable string name for this service instance</td>
</tr>
<tr>
<td>regionsRef</td>
<td>Array of strings</td>
<td>Yes</td>
<td>.<em>\smregion..</em></td>
<td>A comma separated list of regions to which this Service Instance applies. It can be an empty array.</td>
</tr>
<tr>
<td>imageManifestRef</td>
<td>String</td>
<td>No</td>
<td>.<em>\smimagemanifest..</em></td>
<td>^$</td>
</tr>
<tr>
<td>templateRef</td>
<td>String</td>
<td>Yes</td>
<td>.<em>\smservicetemplate..</em></td>
<td>ID of the servicetemplate document For example: smbbase.smservicetemplate.vod_1</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td></td>
<td>The array may have 0+ elements and any number of key value pair objects in the array are allowed. Repeated key names are allowed. Keys are case sensitive in applications; lower case is preferred.</td>
</tr>
<tr>
<td>customConfigs</td>
<td>Array</td>
<td>true</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

customConfigs Item Properties

<table>
<thead>
<tr>
<th>name</th>
<th>Type</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>true</td>
</tr>
</tbody>
</table>

---

Property Name: title
Type: String
Required: Yes
Pattern/Value: 
Description: Customer provided short displayable string name for this service instance.

Property Name: regionsRef
Type: Array of strings
Required: Yes
Pattern/Value: .*\smregion\..*
Description: A comma separated list of regions to which this Service Instance applies. It can be an empty array.

Property Name: imageManifestRef
Type: String
Required: No
Pattern/Value: .*\smimagemanifest\..*|^$ For example: smtenant_0.smimagemanifest.0
Description: ID of the imageManifest document.

Property Name: templateRef
Type: String
Required: Yes
Pattern/Value: .*\smservicetemplate\..* For example: smbbase.smservicetemplate.vod_1
Description: ID of the servicetemplate document.

Property Name: description
Type: String
Required: No

Property Name: customConfigs
Type: Array
Required: true
Pattern/Value: Any number of key value pair objects in the array are allowed. Repeated key names are allowed. Keys are case sensitive in applications; lower case is preferred.

---
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "id": "sm-service-instance-cos-0-1.capture-endpoint.ce1",
  "name": "ce1",
  "type": "capture-endpoint",
  "externalId": "/v2/service-instances/cos-0-1/capture-endpoints/ce1",
  "transactionId": "eac2b584-d159-4b63-87b4-a4cc4c851f5b",
  "properties": {
    "storages": [
      {
        "storageRef": "smregion_0.smcosnodecluster.cluster.ca.01",
        "authProfileRef": "smttenant_0.smauthprofile.auth-1"
      }
    ],
    "slaType": "resource",
    "resourceSLA": {
      "minNode": 1,
      "desiredNode": 3,
      "maxNode": 6,
      "maxStorage": 0
    },
    "regionRef": "smttenant_system.smregion.region-0",
    "description": "A sample capture endpoint for COS Service"
  }
}

Listing One Service Endpoint

GET
https://SM_FQDN:8043/v2/service-instances/service-instance_name/capture-endpoints/endpt_name

Creating a Service Endpoint

POST
https://SM_FQDN:8043/v2/service-instances/service-instance_name/capture-endpoints/endpt_name

Example:

- Request curl:
  
curl -k -H "Content-Type: application/json" -X POST --data @body.json
  https://10.10.150.22:8043/v2/capture-endpoints/ce2

- @body.json file format:

  
  {
    "properties": {
      "storages": [
        {
          "storageRef": "smregion_0.smcosnodecluster.cluster.ca.01",
          "authProfileRef": "smttenant_0.smauthprofile.auth-1"
        }
      ],
      "slaType": "resource",
      "resourceSLA": {
        "minNode": 1,
        "desiredNode": 3,
        "maxNode": 6,
        "maxStorage": 0
      }
    }
  

Cisco Cloud Object Storage Release 2.1.1 API Guide


```
{
  "regionRef": "smtenant_system.smregion.region-0",
  "description": "A capture endpoint for COS Service"
}
```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slaType</td>
<td>enum</td>
<td>Yes</td>
<td></td>
<td>The slaType can be one of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• resource</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• capacity</td>
</tr>
<tr>
<td>resourceSLA</td>
<td>Object</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>capacitySLA</td>
<td>Object</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>regionRef</td>
<td>String</td>
<td>Yes</td>
<td>.<em>.smregion..</em></td>
<td>ID of the region document</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For example:</td>
<td>smtenant_system.smregion.region-0</td>
</tr>
<tr>
<td>defaultStorageRef</td>
<td>String</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>storages</td>
<td>Array</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>storages Item Properties</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>storageRef</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>ID of a NASStore or CosNodeCluster</td>
</tr>
<tr>
<td>authProfileRef</td>
<td>String</td>
<td>No</td>
<td></td>
<td>Required if storageRef is CosNodeCluster</td>
</tr>
</tbody>
</table>

**Deleting a Service Endpoint**

DELETE

https://SM_FQDN:8043/v2/serviceinstances/serviceinstance_name/captureendpoints/endpt_name

**Updating a Service Endpoint**

PUT

https://SM_FQDN:8043/v2/serviceinstances/serviceinstance_name/captureendpoints/endpt_name

**Example:**

- Request curl:
  ```
  curl -k -H "Content-Type: application/json" -X PUT --data @body.json
  https://10.10.150.22:8043/v2/captureendpoints/ce2
  ```

- @body.json file format:
  ```
  {
  ```
Listing, Creating, Deleting, and Modifying Asset Workflow Templates

An AssetWorkflowTemplate is a driving object to create assets using the specified resources, endpoints, policies, and templates. A Service Instance can have zero or more AssetWorkflowTemplates.

---

**Property Name** | **Type** | **Required** | **Pattern/Value** | **Description**
--- | --- | --- | --- | ---
description | String | No | | |
slaType | enum | Yes | The slaType can be one of the following:
| resource |
capacity | |
resourceSLA | Object | No | | |
capacitySLA | Object | No | | |
regionRef | String | Yes | .*\.smregion\..* For example: smtenant_system.smregion.region-0 | ID of the region document |
defaultStorageRef | String | No | | |
storages | Array | No | | |
storages Item Properties
--- | --- | --- | --- | ---
storageRef | String | Yes | | ID of a NASStore or CosNodeCluster |
authProfileRef | String | No | | Required if storageRef is CosNodeCluster |
The template has a state of enable or disable. When the state is enabled, the assets are ingested and published. When the state is disabled, there are no active assets being ingested or published. For Live, that means no capture, indexing or playout when the state is disabled. For VOD, there are not active ingests, indexing and playout as well.

Any Storage object referenced in AssetWorkflowTemplate overrides the capture endpoint definition.

For VoD service, when the VOD source asset is a CIF asset, it is possible to have the Playback Engine retrieve the asset from the Capture engine over HTTP interface, or directly from the Capture Endpoint Storage (NAS). To select the latter option, a property `playbackFromStorageAllowed` should be set to `true`.

### Listing the Asset Workflow Templates

To retrieve a list of all Asset Workflow templates, use the following HTTP GET request:

**GET** `https://SM_FQDN:8043/v2/serviceinstances/serviceinstance_name/assetworkflowtemplates`

This command returns a list of Asset Workflow template objects:

```json
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
{
  "id": "smserviceinstance_cos_0_1.smassetworkflowtemplate.awt-cos",
  "name": "awt-cos",
  "type": "assetworkflowtemplates",
  "externalId": "/v2/serviceinstances/cos-0-1/assetworkflowtemplates/awt-cos",
  "transactionId": "f12d305c-c52a-4f9e-a006-cb136193df1b",
  "properties": {
    "captureEndpointRef": "smserviceinstance_cos_0_1.smcaptureep.ce1",
    "assetRedundancyPolicyRef": "smtenant_0.smassetredundancypolicy.redundancy-pol-cos",
    "state": "enabled"
  }
}
```

### Listing One Asset Workflow Templates

**GET** `https://SM_FQDN:8043/v2/serviceinstances/serviceinstance_name/assetworkflowtemplates/workflow_name`

### Creating a Workflow Templates

**POST** `https://SM_FQDN:8043/v2/serviceinstances/serviceinstance_name/assetworkflowtemplates/workflow_name`

**Example:**

- **Request curl:**
  
  ```bash
curl -k -H "Content-Type: application/json" -X POST --data @body.json https://10.10.150.22:8043/v2/serviceinstances/cos-0-1/assetworkflowtemplates/awt-cos
  
  @body.json file format:
  ```
  ```json
  {
    "properties": {
      "captureEndpointRef": "smserviceinstance_cos_0_1.smcaptureep.ce2",
      "assetRedundancyPolicyRef": "smtenant_0.smassetredundancypolicy.redundancy-pol-cos",
      "state": "enabled"
    }
  }
  ```
### Deleting a Workflow Templates

**DELETE** `https://SM_FQDN:8043/v2/serviceinstances/serviceinstance_name/assetworkflowtemplates/workflow_name`

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>enum</td>
<td>No</td>
<td>type can be one of the following values:</td>
<td>Optional for COS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- livecaptureplayback</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- vodcaptureplayback</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- cdvrcaptureplayback</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- cdvrcapture</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- livecapture</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- playback</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- playbackciforigindynamic</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- asperavodcaptureplayback</td>
<td></td>
</tr>
<tr>
<td>captureEndpointRef</td>
<td>String</td>
<td>No</td>
<td>.<em>.smcaptureendpoint..</em></td>
<td>^$</td>
</tr>
<tr>
<td>playbackEndpointRef</td>
<td>String</td>
<td>No</td>
<td>.<em>.smplaybackendpoint..</em></td>
<td>^$</td>
</tr>
<tr>
<td>assetLifecyclePolicyRef</td>
<td>String</td>
<td>No</td>
<td>.<em>.smassetlifecyclepolicy..</em></td>
<td>^$</td>
</tr>
<tr>
<td>assetRedundancyPolicyRef</td>
<td>String</td>
<td>No</td>
<td>.<em>.smassetredundancypolicy..</em></td>
<td>^$</td>
</tr>
<tr>
<td>assetLineupRef</td>
<td>String</td>
<td>No</td>
<td></td>
<td>ID of ChannelLineup for Live service, or NASMediaSource for VOD, optional for COS</td>
</tr>
<tr>
<td>mediaSourceRef</td>
<td>String</td>
<td>No</td>
<td></td>
<td>Ref to media source</td>
</tr>
<tr>
<td>StorageRef</td>
<td>String</td>
<td>No</td>
<td></td>
<td>Optional. Overrides capture endpoint defaultStorage.</td>
</tr>
<tr>
<td>esamProfileRef</td>
<td>String</td>
<td>No</td>
<td>.<em>.smesamprofile..</em></td>
<td>^$</td>
</tr>
<tr>
<td>state</td>
<td>enum</td>
<td>Yes</td>
<td>state can be either enabled or disabled</td>
<td></td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>assetTemplates</td>
<td>Array</td>
<td>No</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**assetTemplates Item Properties**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abrPublishTemplateRef</td>
<td>String</td>
<td>Yes</td>
<td>.<em>.smpublishtemplate..</em></td>
<td>ID of a ABRPublishTemplate</td>
</tr>
</tbody>
</table>

---

**Deleting a Workflow Templates**
Updating a Workflow Templates

PUT https://SM_FQDN:8043/v2/serviceinstances/serviceinstance_name/assetworkflowtemplates/workflow_name

Example:
- Request curl:
  curl -k -H "Content-Type: application/json" -X PUT --data @body.json

- @body.json file format:
  ```json
  {
    'properties': {
      'captureEndpointRef': "smserviceinstance_cos_0_1.smcaptureep.ce2",
      'assetRedundancyPolicyRef': "smtenant_0.smassetredundancypolicy.redundancy-pol-cos",
      'state': "enabled"
    }
  }
  ```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Required</th>
<th>Pattern/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>enum</td>
<td>No</td>
<td>type can be one of the following values:</td>
<td>Optional for COS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- livecaptureplayback</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- vodcaptureplayback</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- cdvrcaptureplayback</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- cdvrcapture</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- livecapture</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- playback</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- playbackciforigindynamic</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- asperavodcaptureplayback</td>
<td></td>
</tr>
<tr>
<td>captureEndpointRef</td>
<td>String</td>
<td>No</td>
<td>.<em>.smcaptureep..</em>[^$]</td>
<td>ID of CaptureEndPoint used by the AWT. Optional for types playbackciforigindynamic.</td>
</tr>
<tr>
<td>playbackEndpointRef</td>
<td>String</td>
<td>No</td>
<td>.<em>.smplaybackep..</em>[^$]</td>
<td>ID of PlaybackEndPoint used by the AWT. Optional for COS.</td>
</tr>
<tr>
<td>assetLifecyclePolicyRef</td>
<td>String</td>
<td>No</td>
<td>.<em>.smassetlifecyclepolicy..</em>[^$]</td>
<td>ID of AssetLifecyclePolicy used by the AWT</td>
</tr>
<tr>
<td>assetRedundancyPolicyRef</td>
<td>String</td>
<td>No</td>
<td>.<em>.smassetredundancypolicy..</em>[^$]</td>
<td>ID of AssetRedundancyPolicy used by the AWT</td>
</tr>
<tr>
<td>assetLineupRef</td>
<td>String</td>
<td>No</td>
<td></td>
<td>ID of ChannelLineup for Live service, or NASMediaSource for VOD, optional for COS</td>
</tr>
</tbody>
</table>
Viewing COS Node Status

NodeStatus is a read-only object that provides the status of a Node object. It provides the state, fault detail (if any), linkage to a Service Instance and endpoint, the application using it, and all interface details.

Viewing the Status of COS Nodes

To view the status of the COS nodes, use the following HTTP GET request:

```
GET https://SM_FQDN:8043/v2/regions/regionName/nodestatuses
```

This command returns the status of COS node objects:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "id": "smregion_0_status.mmnodestatus.173926472",
  "name": "173926472",
  "type": "nodestatuses",
  "externalId": "v2/regions/region-0/nodestatuses/173926472",
  "transactionId": "785630ca-5a7c-4070-b918-d0ba18f7e3a2",
  "properties": {
    "hostname": "10.93.232.72",
    "serviceInstanceRef": "smtenant_0.smsserviceinstance.cos-0-1",
    "endpointRef": "smserviceinstance_cos_0_1.smcaptureep.ce1",
    "state": "inuse",
    "faultStatus": "Critical",
    "faultDetail": "-- All 1 network interfaces are down on the COS Node
-- All 0 disks are down on the COS Node
-- 2/5 services are down on the COS Node.
-- Service: Cisco Cache Server is down
-- Service: Cassandra is down",
    "lastModifiedTime": "2014-12-19T18:08:04.970Z",
    "interfaces": [
      {
        "interface": "eth2",
        "type": "service",
        "inet": "10.93.232.153",
        "netmask": "255.255.255.224",
        "inetv6": "",
        "gw": "10.93.232.129",
      }
    ]
  }
}
```
Viewing Events for Nodes

Event is a read-only object that provides a view into a state change inside COS. It provides the insight into progress of tasks, warnings, and errors. Users can use this information to see the system reaction to any configuration updates and runtime checks for events by severity.

Viewing the Events for Nodes in a Region

To view the events for COS nodes in a specific region, use the following HTTP GET request:

GET https://SM_FQDN:8043/v2/regions/regionName/events

This command returns the events for COS node objects in a region:

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
{
  "id": "smregion_0_events.smevent.pam-adb6ac90-8660-11e4-bccb-73eb6a8f2a1f",
  "name": "pam-adb6ac90-8660-11e4-bccb-73eb6a8f2a1f",
  "type": "events",
  "externalId": "/v2/regions/region-0/events/pam-adb6ac90-8660-11e4-bccb-73eb6a8f2a1f",
  "transactionId": "35193284-29f4-4fcf-9bf5-e036235b992c",
  "properties": {
    "source": {
      "imgTag": "pam",
      "personality": "control"
    },
    "type": "application",
    "subType": "Interface",
    "event": "DnsUpdate",
    "detailText": "deleteRecord {u'ip': u'10.93.232.153', u'domain': u'mos.npi.cds.cisco.com', u'name': u'', u'host': u'auth01', u'record': u'A'} Completed.",
    "eventTime": "2014-12-18T02:50:50.729Z",
    "eventsDropped": 0,
    "severity": "info",
    "location": {
      "processName": "",
      "processId": "",
      "timestamp": "2014-12-18T02:50:50.729Z"
    }
  }
}
Viewing the Events for Nodes in a Service Instance

To view the events for COS nodes in a Service Instance, use the following HTTP GET request:

**GET https://SM_FQDN:8043/v2/serviceinstances/serviceinstance_name/events**

This command returns the events for COS node objects in a region:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

[{
  "id": "sm/serviceinstance_cos_0_1_events.smevent.cos-aic-96b1fe90-8684-11e4-bfb1-51c0a6dc6375",
  "name": "cos-aic-96b1fe90-8684-11e4-bfb1-51c0a6dc6375",
  "type": "events",
  "externalId": "/v2/serviceinstances/cos-0-1/events/cos-aic-96b1fe90-8684-11e4-bfb1-51c0a6dc6375",
  "transactionId": "fd7d7c52-94f8-4df8-94f8-4543-9bfe-5d8bafbf367",
  "properties": {
    "source": {
      "imgTag": "cos-aic",
      "personality": "linux",
      "node": "COS-PAM.mos.npi.cds.cisco.com"
    },
    "type": "node",
    "subType": "Accessibility",
    "event": "COSNode.Down",
    "detailText": "COS Node 173926472 has not reported status for last 14581 secs",
    "eventTime": "2014-12-18T07:07:55.001Z",
    "eventsDropped": 0,
    "severity": "critical",
    "location": {
      "ipAddr": "127.0.0.1",
      "processName": ",",
      "processId": "1706"
    }
  }
}]
```

**Event Query Filters:**

The Events can be far too large to return in one query. When all events are requested, only the last five minutes of events are returned. In the request, you can filter the events to return based on specific time range, severity, and type.

Add the following parameters in the query to limit the output:

- **startTime**: Return events that have a timestamp after this time (must be used with endTime)
- **endTime**: Return events that have a timestamp of events before this time (must be used with startTime)
- **severity**: Return events that have the specific severity, based on a string to match. The valid values are:
  - critical
  - major
Warning

- **type**: Return events that have the specified type class. The valid type values are:
  - Node
  - Application
  - Ext-Resource
  - High-Availability
  - Config

For example:

```plaintext
GET /v2/serviceinstances/instanceName/events?startTime=2014-03-20T14:00:00.000Z&endTime=2014-03-20T14:10:00.000Z
```

To limit the returned events to a sub-range, specify the time range and add a Range header in the HTTP request:

- **Range**: items=1-100, where the range can be:
  - `-n` (retrieve from `n` and above, `n` is zero based)
  - `-m` (retrieve the last `m` items)
  - `n-m` (retrieve from `n` to `m` items)

- The range query is within the bound of the startTime and endTime parameters, if specified. If startTime and endTime are absent, the range is within the bound of the entire events that are available. On a single request, a maximum of 500 events can be returned.

### Viewing Application Status for the COS Instance

Each Service Instance configuration instantiated contains a certain combination of applications for each endpoint in the Service Instance. Each Application reports its status (via an AppStatus object) in terms of meeting the endpoint SLA and status. In addition, the AppStatus object includes a summary of the nodes it is using, and the number of open events, based on the severity (critical, major, and warning).

AppStatus is a read-only object that provides a view into the application status for each endpoint in a Service Instance.

To view the status for a COS Service Instance, use the following HTTP GET request:

```plaintext
GET https://SM_FQDN:8043/v2/serviceinstances/cosinstance_name/appstatuses/cos
```

This command returns the status for a COS Service Instance:

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

```
{
  "id": "smserviceinstance_cos_0_1_status.smappstatus.cos",
  "name": "cos",
  "type": "appstatuses",
  "externalId": "/v2/serviceinstances/cos-0-1/appstatuses/cos",
  "transactionId": "d53c4141-14a4-446c-97f3-1e98c06793d7",
  "properties": {
    "appName": "cos",
    "endpointRef": "smserviceinstance_cos_0_1.smcapture.ep cél",
  }
}
```
"slaType": "resource",
"slaStatus": {
  "nodesInUse": 1,
  "bandwidthStatus": "unavailable",
  "sessionsStatus": "unavailable",
  "storageStatus": "unavailable",
  "nodeStatus": "normal"
},
"eventsOpen": {
  "critical": 0,
  "major": 0,
  "warning": 0
}
]
Swauth API

This chapter describes the subset of the OpenStack Swauth API that is implemented for the COS authentication service. The Swauth API uses the Authentication FQDN that is assigned to the COS cluster.

Note

The COS cluster is assigned an Authentication FQDN (used with the Swauth API) and a Storage FQDN (used with the Swift API). Currently the Authentication FQDN and the Storage FQDN must be the same, for example, auth01.cos.acme.com.

Note

The Swauth service uses a special built-in user and key to configure the service itself. This user is known as the “super-admin”. The default Swauth super-admin user name is “.super_admin” and the default super-admin key is “rootroot”.

Listing Accounts

To retrieve a list of existing accounts for the reseller or super admin, use the following HTTP GET request:

GET /auth/v2/ HTTP/1.1

Required Headers

- X-Auth-Admin-User — the accounts for this particular user are listed in the response.
- X-Auth-Admin-Key

Response Status Codes

- 200 – Success
- 401 – Invalid X-Auth-Admin-User/X-Auth-Admin-Key
- 5xx – Internal error

Sample Response
Retrieving Account Details

To retrieve the details of an account, use the following HTTP GET request:

```
GET /auth/v2/<account> HTTP/1.1
```

A JSON dictionary of account_ids, services and users is returned.

- The account_id is the value used in creating service accounts.
- The services value is a dict that represents valid storage cluster endpoints, and identifies the default endpoint.
- The users value is a list of dicts, each dict representing a user and currently containing the key name.

**Required Headers**

- X-Auth-Admin-User
- X-Auth-Admin-Key

**Response Status Codes**

- 200 – Success
- 403 – Invalid X-Auth-Admin-User/X-Auth-Admin-Key
- 5xx – Internal error

**Sample Response**

```
HTTP/1.1 200 OK

{
    "services":
    {
        "storage":
        {
            "default": "local",
            "local": "https://<storage endpoint>/v1/<account_id>"
        }
    },
    "account_id": "<account_id>",
    "users": [
        { "name": "user1" },
        { "name": "user2" }
    ]
}
```
Creating an Account

To create a new authentication account, use the following HTTP PUT request:

**Note**
To create a new account, you must be a super admin or a reseller admin.

```
PUT /auth/v2/<account> HTTP/1.1
```

An authentication account allows you to manage a collection of related users, groups and service catalogs.

**Choosing the Account Name**
- The name must not begin with a period (.).
- The name must not include a colon (:).
- The name must not exceed 256 bytes in length.

**Required Request Headers**
- X-Auth-Admin-User: <admin user name>

**Note**
The <admin user name> for users other than the super admin must be of the form <account-name>: <user-name>.

- X-Auth-Admin-Key: <admin user password>

**Optional Request Header**
- X-Account-Suffix: <service account suffix>
  - When an account is created, a new UUID4 with the reseller prefix form the account ID. Using this header, you can replace the UUID4 part of the ID with the <service account suffix>.
  - The <service account suffix> must not exceed 251 bytes in length.

**Response Status Codes**
The response status code is one of the following:
- 201 – Account was created.
- 202 – Account already exists.
- 400 – Account name is invalid.
- 401 – Invalid X-Auth-Admin-User and/or X-Auth-Admin-Key.
- 403 – User not authorized to create an account.
- 5xx – Internal error.
Deleting an Account

To delete an authentication account, use the following HTTP DELETE request:

```
Note
- To delete an account, you must be a super admin or a reseller admin.
- The account should not have users, containers and/or objects.

DELETE /auth/v2/<account> HTTP/1.1
```

**Required Request Headers**
- X-Auth-Admin-User: <admin user name>

```
Note
The <admin user name> for users other than the super admin must be of the form <account-name: user-name>.
```
- X-Auth-Admin-Key: <admin user password>

**Response Status Codes**
The response status code is one of the following:
- 204 – Account was deleted.
- 401 – Invalid X-Auth-Admin-User and/or X-Auth-Admin-Key.
- 403 – User not authorized to delete the account.
- 404 – Account not found.
- 409 – Account is not empty.
- 5xx – Internal error.

Creating or Updating a User

To create a user who can access storage services, use the following HTTP PUT request:

```
Note
- Only the super admin can create reseller admin users.
- An account admin, an authorized reseller admin, or the site admin can create regular and admin users.
- A reseller admin can create a user in any account.
- Admins can create users only in their respective accounts.

PUT /auth/v2/<account>/<user> HTTP/1.1
```

Choosing the User Name
• The name must not begin with a period (.)
• The name must not exceed 256 bytes in length.

**Required Request Headers**

• X-Auth-Admin-User: <admin user name>

**Note**
The <admin user name> for users other than the super admin must be of the form <account-name>: <user-name>.

• X-Auth-Admin-Key: <admin user password>

**Optional Request Headers**

• X-Auth-User-Key: <new user password>
  – This header allows you to specify the password for a new user.
  – The header can also be used by existing users to change their password. Here, the X-Auth-Admin-Key header must have their current password, and the X-Auth-User-Key, the new password.

• X-Auth-User-Admin: true
  – This header allows you to grant admin privileges to the user being created.

• X-Auth-User-Reseller-Admin: true
  – This header allows you to grant reseller admin privileges to the user being created.

**Note**
The admin privileges of an existing user cannot be modified by using the X-Auth-User-Admin: true and X-Auth-User-Reseller-Admin: true headers.

**Response Status Codes**
The response status code is one of the following:

• 201 – Success.
• 400 – Invalid user name.
• 401 – Invalid X-Auth-Admin-User and/or X-Auth-Admin-Key.
• 403 – User not authorized to perform the operation.
• 404 – Account does not exist.
• 5xx – Internal error.

### Retrieving User Details

To retrieve the details of a user, use the following HTTP GET request:

```
GET /auth/v2/<account>/<user> HTTP/1.1
```
A JSON dictionary of the following format is returned:

```json
{
  "groups": [
    {
      "name": "<act>:<usr>",
      # The first group is a unique user identifier
    },
    {
      "name": "<account>",
      # The second group is the auth account name
    },
    {
      "name": "<additional-group>",
      # There may be additional groups, .admin being a special group indicating an account admin and .reseller_admin indicating a reseller admin.
    }
  ],
  "auth": "<auth-type>:<key>"
  # The auth-type and key for the user; currently only plaintext and sha1 are implemented as auth types.
}
```

### Required Headers

- **X-Auth-Admin-User**
  - This header must be set to super-admin to retrieve details of reseller-admin users.
  - This header must be set to super-admin or reseller-admin to retrieve details of account-admin users.
  - This header must be account-admin to retrieve details of unprivileged users.
- **X-Auth-Admin-Key**

### Response Status Codes

- **200** – Success
- **400** – The account or user name starts with a "."
- **401** – Invalid X-Auth-Admin-User/X-Auth-Admin-Key
- **403** – Retrieval of the requested user barred by admin user.
- **404** – Unknown account or user
- **5xx** – Internal error

### Deleting a User

To remove a user, use the following HTTP DELETE request:

**Note**
- A super admin or a reseller admin can delete users from any account.
Account admins can only delete users from the accounts they administer.

DELETE /auth/v2/<account>/<user> HTTP/1.1

Required Request Headers
- X-Auth-Admin-User: <admin user name>
- X-Auth-Admin-Key: <admin user password>

Note: The <admin user name> for users other than the super admin must be of the form <account-name: user-name>.

Response Status Codes
The response status code is one of the following:
- 200 – User deleted.
- 400 – Invalid user name.
- 403 – User not authorized to perform the operation.
- 404 – Account/user does not exist.
- 5xx – Internal error.

Creating or Updating Account Service Endpoints

To create new service endpoints or update existing ones, use the following HTTP POST request on the pseudo-user `.services`:

POST /auth/v2/<account>/.services HTTP/1.0

The request must also contain the a JSON dictionary of the following format:
{"service_name": {"end_point_name": "end_point_value"}}

- Multiple services and multiple endpoints can be specified in a single request.
- New services and endpoints will be added to the existing set of services and endpoints, respectively.
- If the service specified exists, new end points will be linked to it.
- If the endpoint specified exists, its value is updated.

The updated services dictionary will be returned on success.
Getting an Authentication Token

To get an authentication token, use the following HTTP GET request:

GET /auth/v1.0 HTTP/1.1

Required Request Headers

- X-Auth-User: <user name>

Note The <user name> for users other than the super admin must be of the form
<account-name>: <user-name>.

- X-Auth-Key: <user password>

Response Headers

On successful authentication, the response has the following headers:

- X-Auth-Token
  - This header has the token that is to be used with the Swift APIs used in container and object
    management.

- X-Storage-URL
  - This header has the URL of the default storage cluster.

Response Body

On successful authentication, the response body is set to the account’s services JSON object as shown below.

```json
{"storage": {"# Represents the Swift storage service end points
```
"default": "cluster1", # Indicates which cluster is the default
"cluster1": "<URL to use with Swift>",
# A Swift cluster that can be used with this account,
# "cluster1" is the name of the cluster which is usually a
# location indicator (like "dfw" for a datacenter region).
"cluster2": "<URL to use with Swift>",
# Another Swift cluster that can be used with this account,
# there will always be at least one Swift cluster to use or
# this whole "storage" dict won't be included at all.
"servers": {
# Represents the Nova server service end points
# Expected to be similar to the "storage" dict, but not
# implemented yet.
},
# Possibly other service dicts, not implemented yet.
Swift API

This chapter describes the subset of the OpenStack Swift API that is implemented for COS. The Swift API uses the Storage FQDN that is assigned to the COS cluster.

Note

The COS cluster is assigned an Authentication FQDN (used with the Swauth API) and a Storage FQDN (used with the Swift API). Currently the Authentication FQDN and the Storage FQDN must be the same, for example, auth01.cos.acme.com.

Listing Containers

To retrieve a list of existing storage containers, use the following HTTP GET request:

Note

To make this request, you must be an account admin or a reseller admin.

GET /v1/<account>[/?<param>=<value>[/&<param>=<value>]] HTTP/1.1

The names of the containers in the list are sorted based on a binary comparison of the UTF-8 encoded container names.

Required Request Header

X-Auth-Token: <user token>

Optional Query Parameters

The following parameters can be used in the query:

• limit — Specifies the maximum number of results to be retrieved.
• marker — Retrieves container names whose characters have a greater Unicode alphabetical value than those of the specified string.
• end_marker — Retrieves container names whose characters have a lower Unicode alphabetical value than those of the specified string.
• prefix — Retrieves container names beginning with the specified characters.
• delimiter — Retrieves container names that do not have the specified character, except in the prefix, if any.
• format — Specifies either json or xml as the format of the serialized response.

### Listing Objects

To list the objects in the storage container, use the following HTTP GET request:

```
GET /v1/<account>/<container>[?<param>=<value>[&<parm>=<value>]] HTTP/1.1
```

The names of the objects in the list are sorted based on a binary comparison of the UTF-8 encoded object names.

#### Optional Query Parameters

The following parameters can be used in the query:

• limit — Specifies the maximum number of results to be retrieved.
• marker — Retrieves object names whose characters have a greater Unicode alphabetical value than those of the specified string.
• end_marker — Retrieves object names whose characters have a lower Unicode alphabetical value than those of the specified string.
• prefix — Retrieves object names beginning with the specified characters.
• delimiter — Retrieves object names that do not have the specified character, except in the prefix, if any.
• path — Retrieves names of objects nested in the specified path.
• format — Specifies either json or xml as the format of the serialized response.

If a response format is not specified as a query parameter, a list of object names is returned in the response body, one name per line.

#### Response Status Codes

The response status code is one of the following:
Creating a Container

To create a storage container, use the following HTTP PUT request:

```
PUT /v1/<account>/<container> HTTP/1.1
```

The name of the container must adhere to the following restrictions:
- The name cannot include the forward slash (/) character or the encoded forward slash character (%2F or %2f).
- The name should not exceed 256 bytes when it is encoded in URL.

Required Request Header
- X-Auth-Token: <user token>

Optional Request Headers
- X-Container-Read: <read acl>
- X-Container-Write: <write acl>

Assigning Custom Attributes
To assign custom attributes to a storage container, include additional HTTP headers in the HTTP PUT request shown above. The additional headers should be of the following form:
```
X-Container-Meta-<attribute name>: <attribute value>
```

Response Status Codes
The response status code is one of the following:
- 201 (Created) — The container was created.
- 202 (Accepted) — The container already exists.
- 400 (Bad Request) — Invalid container name.
- 401 (Unauthorized) — The user token is missing or invalid.
- 403 (Forbidden) — The user does not have permission to create the container.
- 404 (Not Found) — The storage URL references a non-existent account.
Deleting a Container

To permanently remove a storage container, use the following HTTP DELETE request:

Note
To make this request, you must be an account admin, a reseller admin, or the super admin.

DELETE /v1/<account>/<container> HTTP/1.

Note
Only empty storage containers can be deleted.

Required Request Header
X-Auth-Token: <user token>

Response Status Codes
The response status code is one of the following:
- 204 (No Content) — The container was deleted.
- 400 (Bad Request) — Invalid container name.
- 401 (Unauthorized) — The user token is missing or invalid.
- 403 (Forbidden) — The user does not have permission to delete the container.
- 404 (Not Found) — The storage URL references a non-existent account.
- 409 (Conflict) — The container is not empty.

Retrieving an Object

To retrieve the data of an object, use the following HTTP GET request:

Note
To make this request, you must be an account admin or COS reseller-admin; or if a read-ACL has been specified for the container, either you are assigned a role listed in the ACL, or * is specified as a role, permitting anonymous access.

GET /v1/<account>/<container>/<object> HTTP/1.1

Note
You may retrieve data from an object while it is being created. If you do, the response will have the X-Object-Is-Dynamic: yes header.

Required Request Header
X-Auth-Token: <user token>
If the container read-ACL permits anonymous access, this header is not required.

Optional Request Headers

- **X-Follow-Redirect: true**
  - If the request has this header, the COS node may respond with a 307 (Temporary Redirect) code and include a Location response header having the URL at which the client should retry the request.
  - If the request does not have this header, or the value of the header is not true, the COS node receiving the request will respond with the requested object.

  **Note** The X-Follow-Redirect request header is a COS extension.

- **X-Transfer-Rate: <bits-per-second>**
  - This header specifies a transfer rate in decimal bits per second.
  - Valid range of values for this header is 400000 to 50000000, that is, 400 Kbps to 50 Mbps.
  - If the header is not included, data is transferred at the best-effort rate that does not delay other transfers which were requested along with an X-Transfer-Rate header.
  - A transfer rate of 0 is valid and indicates a best-effort transfer of data.

  **Note** The X-Transfer-Rate request header is a COS extension.

- **X-Transfer-Delay: <delta-time-in-milliseconds>**
  - This header specifies a signed delay in milliseconds.
  - If the delay is positive, COS waits for the time interval specified before starting the transfer of object data at the requested transfer rate. The positive delay can have a maximum value of 30 seconds.
  - A negative delay indicates that the client wants to use an elasticity buffer and intends to transmit data from the partially full buffer that receives data from COS at the requested transfer rate. In this case, COS starts transferring the data as soon as the retrieve request is received, and attempts to send data at a rate higher than the requested rate. COS, in essence, tries to match the amount of data that would have been sent to the client if the data transfer had been initiated “delay” seconds before the receipt of the request. The negative delay can have maximum magnitude of four seconds.

  **Note** The X-Transfer-Delay request header is a COS extension.

- **Range: bytes = <byte-range>**
  - To request the transfer of specific portions of the object data, in accordance with the specifications in section 14.35 of RFC 2616, include this header.
  - Only byte-ranges are supported.
  - Multiple byte ranges are supported.
- If the range did not include the entire object, a response status code of 206 (Partial Content) is returned by COS.
- The Partial Content response to a request for multiple non-overlapping ranges of data contains multiple parts in the message body.

- **If-Match: ETag**
  - The object data is retrieved only if the client specified ETag value matches the ETag of the content. Else, 412 (Precondition Failed) is returned.

- **If-None-Match: ETag**
  - 304 (Not Modified) is returned if the client specified ETag value matches the ETag of the content, indicating to the client that the object cached by it has not been modified since.

- **If-Modified-Since: time**
  - 304 (Not Modified) is returned if the client specified time is equal to or later than the last modified time of the object.

- **If-Unmodified-Since: time**
  - The object data is retrieved only if the client specified time is equal to or later than the last modified time of the object. Else, 412 (Precondition Failed) is returned.

### Response Status Codes

A response status code of **2xx** indicates successful completion of the request.

### Response Headers

The response will include one of the following headers:

- **Last Modified** — A time-stamp of when the object was created or modified.
- **ETag** — The hexadecimal representation of the MD5 hash of the object data.
- **Content-Type** — The content type associated with the object when it was created.
- **Content-Length** — The number of bytes in the object.
- **X-Object-Goid** — The global object identifier assigned by COS when the object was created.

**Note**
The X-Object-Goid response header is a COS extension.

- **X-Object-Meta-*** — Custom object attributes of the object.
- **X-Object-Is-Dynamic: yes** — The object being retrieved is being extended. This occurs when you retrieve object data while object creation is in progress. Such a retrieval of the data is useful when a large object is being created and you want to access the data that has been stored, even as more data is being appended to the object. This response header is a COS extension.

**Note**
The X-Object-Is-Dynamic response header is a COS extension.
Creating or Updating An Object

To create or update an object, that is, to write or overwrite an object’s content and metadata, use the following HTTP PUT request:

```
PUT /v1/<account>/<container>/<object> HTTP/1.1
```

**Required Request Headers**

- **X-Auth-Token: <user token>**

  **Note** This header may be excluded if the container write ACL permits anonymous access.

- **Content-Type**
  - If this header is not included, the system will attempt to guess the type of the content based on the object’s name/metadata. If the system is unsuccessful, the Content-Type is set to the default application/octet-stream value.

- **Content-Length/Transfer-Encoding: chunked**
  - Either a valid Content-Length header, stating the size of the object, or a Transfer-Encoding: chunked header, indicating that the data length is encoded in-line at the start of each chunk of the object data sent in the request, must be included.

**Optional Request Headers**

- **ETag**
  - The request may include this header with the value set to the hexadecimal representation of the MD5 hash of the object data.
  - If the ETag value does not match MD5 hash computed by COS, a 422 (Unprocessable Entity) response status code is returned.

- **X-Object-Meta-<attribute name>: <attribute value>**
  - Include this header to set custom attributes for an object.

- **X-Follow-Redirect: true**
  - If this header is included, the COS node may respond with a status code of 307 (Temporary Redirect) and include a Location response header indicating the URL to which the client should address the request.
  - If this header is absent, or the value of the header is not true, the COS node receiving the request will perform the necessary operation.

  **Note** The X-Follow-Redirect request header is a COS extension.

- **Expect: 100-Continue**
  - It is recommended that you include this header in the initial request and omit object content from the body of the request.
Deleting an Object

To permanently remove an object, use the following HTTP DELETE request:

```
DELETE /v1/<account>/<container>/<object> HTTP/1.1
```

Deleting an object removes both the object data and metadata. Any subsequent operations attempted on the object will return a 404 (Not Found) response status code.
Creating or Updating Container Metadata

To create or update custom container attributes, use the following HTTP POST request:

```
POST /v1/<account>/<container> HTTP/1.1
```

The attributes are specified in HTTP headers included in the HTTP POST request. If the attribute exists, its value will be overwritten. Else, a new attribute is created.

**Required Request Header**

- X-Auth-Token: <user token>
- X-Container-Meta-<attribute name>: <attribute value>

**Response Status Code**

The response status code is one of the following:

- 204 (No Content) — The POST operation succeeded
- 400 (Bad Request) — The POST request is not valid
- 401 (Unauthorized) — The user token is missing or invalid
- 403 (Forbidden) — The user does not have permission to modify the container attributes
- 404 (Not Found) — The storage URL references a non-existent account or container
- 5xx — Internal Server Error
Retrieving Container Metadata

To retrieve a container’s metadata to learn its status, use the following HTTP HEAD request:

**Note**
- To make this request, you must be an account admin or a COS reseller admin.
- Or, if a read-ACL has been specified for the container, you must be assigned a role listed in the ACL.
- Or, the '*' role must be included in the ACL, permitting anonymous access.

```plaintext
HEAD /v1/<account>/<container> HTTP/1.1
```

This request can be used against a container to determine the number of objects, and the total byte size of all objects stored in the container.

**Required Request Header**

**Note**
If the container read-ACL permits anonymous access, this header is not required.

```plaintext
X-Auth-Token: <user token>
```

**Response Status Code**

The response status code is one of the following:

- 2xx (Success) — The HEAD operation succeeded
- 400 (Bad Request) — The POST request is not valid
- 401 (Unauthorized) — The user token is missing or invalid
- 403 (Forbidden) — The user does not have permission to modify the container attributes
- 404 (Not Found) — The storage URL references a non-existent account or container
- 5xx — Internal Server Error

**Response Headers**

- **X-Container-Object-Count**
  - The value of this header is the number of objects in the container.
- **X-Container-Bytes-Used**
  - The value of this header is the total byte size of all the objects in the container.
- **X-Container-Meta-<attribute name>: <attribute value>**
  - This header returns the custom attributes of the container.
Deleting Container Metadata

To delete custom container attributes, use the following HTTP POST request:

```
POST /v1/<account>/<container> HTTP/1.1
```

**Note**
To make this request, you must be an account admin or a reseller admin.

**Required Request Headers**
- **X-Auth-Token**: `<user token>`
- **X-Container-Meta-<attribute name>**
  - An empty header of this type without the attribute value can be used to delete the custom attribute named in the header.
- **X-Remove-Container-Meta-<attribute name>: <arbitrary value>**
  - Alternatively, a header of this type can be used to delete the custom attribute named in the header.
  - The arbitrary attribute value is ignored by the system.

**Response Status Code**
The response status code is one of the following:
- **204 (No Content)** — The POST operation succeeded
- **400 (Bad Request)** — The POST request is not valid
- **401 (Unauthorized)** — The user token is missing or invalid
- **403 (Forbidden)** — The user does not have permission to modify the container attributes
- **404 (Not Found)** — The storage URL references a non-existent account or container
- **5XX** — Internal Server Error

Creating or Updating Object Metadata

To create or update custom object attributes, use the following HTTP POST request:

```
POST /v1/<account>/<container>/<object> HTTP/1.1
```

**Note**
- To make this request, you must be an account admin or a COS reseller admin.
- Or, if a write-ACL has been specified for the container, you must be assigned a role listed in the ACL.
- Or, the '* role must be included in the ACL, permitting anonymous access.

```
POST /v1/<account>/<container>/<object> HTTP/1.1
```
Assigning custom attributes to objects enables you to better categorize the objects.

**Required Request Header**

- X-Auth-Token: <user token>

**Note**

If the container write-ACL permits anonymous access, this header is not required.

- X-Object-Meta-<attribute name>: <attribute value>
  - If the attribute exists, its value is updated to that specified in the header. Else, the attribute is created.

**Response Status Code**

The response status code is one of the following:

- 204 (No Content) — The POST operation succeeded
- 400 (Bad Request) — The POST request is not valid
- 401 (Unauthorized) — The user token is missing or invalid
- 403 (Forbidden) — The user does not have permission to modify the container attributes
- 404 (Not Found) — The storage URL references a non-existent account or container
- 5XX — Internal Server Error

---

**Retrieving Object Metadata**

To retrieve an object's metadata, including its custom attributes, use the following HTTP HEAD request:

**Note**

To make this request, you must be an account admin or a COS reseller admin.

- Or, if a read-ACL has been specified for the container, you must be assigned a role listed in the ACL, and the Referrer header in the request must match the ACL referrer pattern.
- Or, the '*' role must be included in the ACL, permitting anonymous access.

**HEAD /v1/<account>/<container>/<object> HTTP/1.**

**Required Request Header**

**Note**

If the container read-ACL permits anonymous access, this header is not required.

- X-Auth-Token: <user token>

**Response Status Code**

The response status code is one of the following:
Deleting Object Metadata

To delete custom object attributes, use the following HTTP POST request:

```
POST /v1/<account> HTTP/1.1
```

**Required Request Headers**

- X-Auth-Token: <user token>
- X-Object-Meta-<attribute name>:
  - An empty header of this type without the attribute value can be used to delete the custom attribute named in the header.
- X-Remove-Object-Meta-<attribute name>: <arbitrary value>
  - Alternatively, a header of this type can be used to delete the custom attribute named in the header.
  - The arbitrary value is ignored by the system.

**Response Status Code**

The response status code is one of the following:

- 204 (No Content) — The POST operation succeeded
Creating or Updating Account Metadata

To create or update custom account attributes, use the following HTTP POST request:

```
POST /v1/<account> HTTP/1.1
```

Required Request Headers
- X-Auth-Token: <user token>
- X-Account-Meta-<attribute name>: <attribute value>
  - If the attribute exists, its value is updated to that specified in the header. Else, the attribute is created.

Response Status Code
The response status code is one of the following:
- 204 (No Content) — The POST operation succeeded
- 400 (Bad Request) — The POST operation is not valid
- 401 (Unauthorized) — The user token is missing or invalid
- 403 (Forbidden) — The user does not have permission to modify the container attributes
- 404 (Not Found) — The storage URL references a non-existent account or container
- 5xx — Internal Server Error

Retrieving Account Metadata

To retrieve account metadata to check the account statistics, use the following HTTP HEAD request:

```
HEAD /v1/<account> HTTP/1.1
```

To perform the operation, you must be an account administrator or a COS reseller admin.
Deleting Account Metadata

Required Request Headers
X-Auth-Token: <user token>

Response Status Codes
The response status code is one of the following:
- 2xx (Success) — The HEAD operation succeeded
- 400 (Bad Request) — The HEAD operation is not valid
- 401 (Unauthorized) — The user token is missing or invalid
- 403 (Forbidden) — The user does not have permission to modify the container attributes
- 404 (Not Found) — The storage URL references a non-existent account
- 5xx — Internal Server Error

Response Headers
- X-Account-Container-Count: <value>
  - The value of the header is the number of containers in the account.
- X-Account-Object-Count: <value>
  - The value of the header is the number of objects in the account.
- X-Account-Bytes-Used: <value>
  - The value of the header is the total number of bytes in COS for the specified account.
- X-Account-Meta-<attribute name>: <attribute value>
  - The header returns custom account attributes and their values.

Deleting Account Metadata

To delete custom account attributes, use the following HTTP POST request:

POST /v1/<account> HTTP/1.1

Required Request Headers
- X-Auth-Token: <user token>
- X-Account-Meta-<attribute name>:
  - An empty header of this type without the attribute value can be used to delete the custom attribute named in the header.
- X-Remove-Account-Meta-<attribute name>: <arbitrary value>
  - Alternatively, a header of this type can be used to delete the custom attribute named in the header.

Note
To perform the operation, you must be an account administrator or a COS reseller admin.
Access Control Lists (ACLs)

By default, to access a storage object, a requester must be an account administrator of the account containing the object. An administrator can modify the access policy for a container and its storage objects by using container access control lists (ACLs). The administrator can specify the read and write access control lists as part of the container metadata. When an ACL is deleted, the default access policy is restored.

An ACL has the following form:

```
[item [, item...]]
```

An ACL item can be one of the following:

- `<account name>`
  - All the users of specified account are granted access to objects in the container.

- `<account name>: <user name>`
  - Users identified by the combination of the specified account and user names are granted access to objects in the container.

- `*`
  - An asterisk permits anonymous access. This option is a COS extension.

Creating or Updating ACLs

To create or update an ACL, use the following HTTP POST request:

```plaintext
POST /v1/<account>/<container> HTTP/1.1
```

**Note**

To perform the operation, you must be an account administrator or a COS reseller admin.
Deleting ACLs

To delete an ACL, use the following HTTP POST request:

Note

To perform the operation, you must be an account administrator or a COS reseller admin.

POST /v1/<account>/<container> HTTP/1.1

Required Request Headers

- X-Auth-Token: <user token>
- X-Container-Read: <read acl>
  - Specify an empty list as the value of this header to delete the read ACL.
- X-Remove-Container-Read: <arbitrary value>
  - Alternatively, a header of this type can be used to delete the read ACL.
  - The arbitrary value is ignored by the system.
- X-Container-Write: <write acl>
  - Specify an empty list as the value of this header to delete the write ACL.
- X-Remove-Container-Write: <arbitrary value>
  - Alternatively, a header of this type can be used to delete the write ACL.
  - The arbitrary value is ignored by the system.
Response Status Code

The response status code is one of the following:

- 204 (No Content) — The POST operation succeeded
- 400 (Bad Request) — The POST request is not valid
- 401 (Unauthorized) — The user token is missing or invalid
- 403 (Forbidden) — The user does not have permission to modify the container attributes
- 404 (Not Found) — The storage URL references a non-existent account or container
- 5xx — Internal Server Error
Example API Calls

This appendix provides some examples performing a Service Manager, Swauth, and Swift API call using curl.

Service Manager API curl Example

To retrieve a list of existing IP Pools:

```bash
curl -v -L -k -X GET https://SM.acme.com:8043/v2/ippools
```

```
* About to connect() to SM.acme.com port 8043 (#0)
* Trying 10.1.1.1... connected
* Connected to SM.acme.com (10.1.1.1) port 8043 (#0)
* Initializing NSS with certpath: sql:/etc/pki/nssdb
* warning: ignoring value of ssl.verifyhost
* skipping SSL peer certificate verification
* NSS: client certificate not found (nickname not specified)
* SSL connection using TLS_RSA_WITH_AES_256_CBC_SHA
* Server certificate:
  * subject: E=support@cisco.com,CN=SM,OU=SPVTG,O=CISCO SYSTEMS,L=MILPITAS,ST=CALIFORNIA,C=US
  * start date: Apr 23 18:39:33 2014 GMT
  * expire date: Apr 20 18:39:33 2024 GMT
  * common name: SM
  * issuer: E=support@cisco.com,CN=SM,OU=SPVTG,O=CISCO SYSTEMS,L=MILPITAS,ST=CALIFORNIA,C=US
> GET /v2/ippools HTTP/1.1
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.13.6.0 zlib/1.2.3 libidn/1.18 libssh2/1.4.2
> Host: SM.acme.com:8043
> Accept: */*
>
< HTTP/1.1 200 OK
< content-length: 2138
< content-type: application/json; charset=utf-8
< x-powered-by: Express
< etag: "1249005776"
< date: Fri, 19 Dec 2014 20:57:03 GMT
< connection: close
<
{
  "id": "smtenant_system.smippool.ippool-1",
  "name": "ippool-1",
  "type": "ippools",
  "externalId": "/v2/ippools/ippool-1",
```
"properties": {
  "description": "A sample ip pool for COS cache interfaces",
  "addrType": "ipv4",
  "networkRef": "smtenant_system.smnetwork.network-a",
  "pool": [
    {
      "rangeStart": "0.0.0.0",
      "rangeEnd": "0.0.0.0",
      "netmask": "255.255.255.0",
      "gw": "0.0.0.0"
    }
  ]
},
"id": "smtenant_system.smippool.2",
"name": "2",
"type": "ippools",
"externalId": "/v2/ippools/2",
"transactionId": "0d6177a1-4e25-4e51-8563-3d65de952baa",
"properties": {
  "description": "COS NPI Pool",
  "addrType": "ipv4",
  "networkRef": "smtenant_system.smnetwork.network-a",
  "pool": [
    {
      "rangeStart": "10.93.232.153",
      "rangeEnd": "10.93.232.153",
      "netmask": "255.255.255.224",
      "gw": "10.93.232.129"
    }
  ]
},
"id": "smtenant_system.smippool.pool-3",
"name": "pool-3",
"type": "ippools",
"externalId": "/v2/ippools/pool-3",
"transactionId": "dc26d15b-37bd-4968-9be4-cbca0b2f0deb",
"properties": {
  "description": "COS node Pool 3",
  "addrType": "ipv4",
  "networkRef": "smtenant_system.smnetwork.network-a",
  "pool": [
    {
      "rangeStart": "10.93.232.155",
      "rangeEnd": "10.93.232.155",
      "netmask": "255.255.255.224",
      "gw": "10.93.232.129"
    }
  ]
},
"id": "smtenant_system.smippool.cos-npi",
"name": "cos-npi",
"type": "ippools",
"externalId": "/v2/ippools/cos-npi",
"transactionId": "2c82ad23-03d2-48e5-b216-1ce6431cddac",
"properties": {
  "description": "cos-npi data interface pool",
  "addrType": "ipv4",
  "networkRef": "smtenant_system.smnetwork.network-a",
}
Swauth API curl Example

To see a list of accounts:

* About to connect() to auth01.cos1.acme.com port 80 (#0)
 * Trying 192.168.1.1... connected
* Connected to auth01.cos1.acme.com (192.168.1.1) port 80 (#0)
> GET /auth/v2 HTTP/1.1
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/1.2.3 libidn/1.18 libssh2/1.4.2
> Host: auth01.cos1.acme.com
> Accept: */*
> X-Auth-Admin-User:gsmith:user1
> X-Auth-Admin-Key:123XYZ
>
< HTTP/1.1 200 OK
< Server: Cisco/Object Store/0.2
< Connection: Keep-Alive
< Date: Fri, 19 Dec 2014 22:18:37 GMT
< Content-Type: application/json; charset=utf-8
< Content-Length: 206
<
> {"accounts": [
  {"name": "abrown"},
  {"name": "cjones"},
  {"name": "cjones"},
  {"name": "gsmith"},
  {"name": "kurt"},
  {"name": "matt"},
  {"name": "michele"},
  {"name": "michael"},
  {"name": "michael"}
  ],
  "version": "1.1.1"
}
* Connection #0 to host auth01.cos1.acme.com left intact
* Closing connection #0

Swift API curl Example

To create a container:

* About to connect() to auth01.cos1.acme.com port 80 (#0)
 * Trying 192.168.1.1... connected
* Connected to auth01.cos1.acme.com (192.168.1.1) port 80 (#0)
Example API Calls

**Swift API curl Example**

```
> PUT /v1/AUTH_msith/mustang HTTP/1.1
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/1.2.3
    libidn/1.18 libssh2/1.4.2
> Host: auth01.cos1.acme.com
> Accept: */*
> X-Auth-Token: AUTH_tka5d2c2898c1e463285f55cadf55f5
>
< HTTP/1.1 201 Created
< Server: Cisco/Object Store/0.2
< Connection: Keep-Alive
< Date: Fri, 12 Dec 2014 01:54:15 GMT
< Content-Length: 0
<
* Connection #0 to host auth01.cos1.acme.com left intact
* Closing connection #0

To confirm the creation of the container:

```
curl -v -L -X GET -H "X-Auth-Token: AUTH_tka5d2c2898c1e463285f55cadf55f5"
http://auth01.cos1.acme.com/v1/AUTH_msmith
```

```
* About to connect() to auth01.cos1.acme.com port 80 (#0)
* Trying 192.168.1.1... connected
* Connected to auth01.cos1.acme.com (192.168.1.1) port 80 (#0)
> GET /v1/AUTH_msmith HTTP/1.1
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/1.2.3
    libidn/1.18 libssh2/1.4.2
> Host: auth01.cos1.acme.com
> Accept: */*
> X-Auth-Token: AUTH_tka5d2c2898c1e463285f55cadf55f5
>
< HTTP/1.1 200 OK
< Server: Cisco/Object Store/0.2
< Connection: Keep-Alive
< Date: Fri, 19 Dec 2014 22:26:58 GMT
< Content-Type: text/plain; charset=utf-8
< Content-Length: 38
< X-Account-Container-Count: 3
< X-Account-Object-Count: 0
< X-Account-Bytes-Used: 0
<
* Connection #0 to host auth01.cos1.acme.com left intact
* Closing connection #0
```