CHAPTER 4

Getting Familiar with the Application Programming Interface Data Types

This module describes the various API data types used in the Service Control Management Suite (SCMS) Service Control Engine (SCE) Subscriber Application Programming Interface (API).

Subscriber ID

Most methods of the SCE Subscriber APIs require the subscriber ID to be used as an input parameter. The Subscriber ID is a string representing a subscriber name or a CM MAC address. This section lists the formatting rules of a subscriber ID.

The subscriber name is *case-sensitive*. It may contain up to 64 characters. All printable characters with an ASCII code between 32 and 126 (inclusive) can be used; except for 34 ("), 39 ('), and 96 (`).

For example:

```
String subID1="john";
String subID2="john@yahoo.com";
```

Information About Network ID Mappings

A network ID is a network identifier that the SCE device relates to a specific subscriber record. A typical example of a network ID mapping is an IP address. Currently, the Cisco Service Control Engine (SCE) supports IP address, IP range, private IP address over VLAN, private IP range over VLAN, and VLAN types of mappings.

The NetworkID class represents various types of subscriber network identification.

The API supports the following subscriber mapping types:

- IP addresses or IP ranges
- Private IP addresses or private IP ranges over VLAN

- VLAN mappings

When using subscriber operations that involve network ID, the caller is requested to provide a NetworkID parameter.

NetworkID class constructors are defined as follows:

```
public NetworkID(String mapping,short mappingType) throws Exception
public NetworkID(String[] mappings,short[] mappingTypes) throws Exception
```

Parameters of the NetworkID constructors are:

- A **java.lang.String** mapping identifier or array of mapping identifiers
- A short mapping type or array of mapping types

When passing arrays, the mappingTypes array must contain either the same number of elements as the mappings array, or a single element.

- Use NetworkID.TYPE_IP or NetworkID.TYPE_VPN constants if the array contains more than one element

Note The NetworkID.TYPE_VLAN and NetworkID.ALL_VLAN_MAPPINGS constants are deprecated and it is recommended to use the NetworkID.TYPE_VPN and NetworkID.ALL_VPN_MAPPINGS constants instead.

- Use NetworkID.ALL_IP_MAPPINGS or NetworkID.ALL_VLAN_MAPPINGS constants when a single array element is used

Specifying IP Address Mapping

The string format of an IP address is the commonly used decimal notation:

```
IP-Address=[0-255].[0-255].[0-255].[0-255]
```

Example:

- 216.109.118.66

  The mapping type of an IP address is provided in the class NetworkID:

  ```
  com.scms.common.NetworkID.TYPE_IP:
  ```

  com.scms.common.NetworkID.ALL_IP_MAPPINGS specifies that all the entries in the mapping identifiers array are IP mappings.

Specifying IP Range Mapping

The string format of an IP range is an IP address in decimal notation and a decimal specifying the number of **1** s in a bit mask: **IP-Range=[0-255].[0-255].[0-255].[0-255]/[0-32]**.

Examples:

- **10.1.1.10/32** is an IP range with a full mask, that is, a regular IP address.
- **10.1.1.0/24** is an IP range with a 24-bit mask, that is, all of the addresses ranging between **10.1.1.0** and **10.1.1.255**.

Note The mapping type of an IP Range is identical to the mapping type of the IP address.

Specifying Private IP Address or Private IP Range over VLAN Mapping

The string format of an IP address and an IP range are described in Specifying IP Address Mapping and Specifying IP Range Mapping. When the network ID mapping uses an IP address or range over VLAN, the string format includes the VLAN number.

Examples:

- **10.1.1.10@1** is an IP address over VLAN1.
- **10.1.1.0/24@2** is an IP range with a 24-bit mask, that is, all of the addresses ranging between **10.1.1.0** and **10.1.1.255** over VLAN2.

Note The mapping type of an IP address or IP range over VLAN is identical to the mapping type of the IP address.

Specifying VLAN Tag Mapping

The string format for VLAN tag mapping is a decimal number in the following range: **[2-2046]**

The **com.scms.common.NetworkID** class provides the VLAN mapping type:

- The mapping type of an IP address is provided in the class NetworkID:

  `com.scms.common.NetworkID.TYPE_VPN:`

- **com.scms.common.NetworkID.ALL_VLAN_MAPPINGS** specifies that all the entries in the mapping identifiers array are VLAN mappings.

Network ID Mappings Examples

Construct NetworkID with a single IP address:

```
NetworkID nid = new NetworkID("1.1.1.1",NetworkID.TYPE_IP)
```

Construct NetworkID with a range of IP addresses:

```
NetworkID nid = new NetworkID("1.1.1.1/24",NetworkID.TYPE_IP)
```

Construct NetworkID with multiple IP addresses:

```
NetworkID nid = new NetworkID(new String[]{"1.1.1.1","2.2.2.2","3.3.3.3"},
NetworkID.ALL_IP_MAPPINGS)
```

Construct NetworkID with a single VLAN address:

```
NetworkID nid = new NetworkID("23",NetworkID.TYPE_VPN)
```

Information About SCA BB Subscriber Policy Profile

The Policy Profile describes the subscriber policy information. A policy profile is generally comprised of two main parts including a statically defined policy that is identified by the policy package and a set of subscriber policy properties that might have a dynamic nature. The package ID identifies the policy package. Most of the rules enforced on the subscriber traffic are derived from the package ID.

Subscriber policy property in SCA BB is a **key-value** pair that affects the way the SCE analyzes and reacts to network traffic generated by the subscriber.

The subscriber properties that are supported by the SCA BB are packageId, monitor, upVlinkId, and downVlinkId. For a description of the subscriber properties, see the *Cisco Service Control Application for Broadband User Guide*.

PolicyProfile Class

The API provides a PolicyProfile class to format subscriber policy profiles required for the API operations.

The following method constructs the PolicyProfile class based on the array of policy properties:

```
public PolicyProfile(String[] policy)
```

Note The encoding of each string within the array **must** be as follows:

```
property_name=property_value
```

The following method allows adding a policy property to the profile according to the format described above:

```
public void addPolicyProperty(String policyProperty)
```

Note This method is not optimized for performance. For best performance results, use the PolicyProfile constructor.

Example:

```
PolicyProfile pp = new PolicyProfile(new String[]{"packageId=22", "monitor=1"})
```

Information About Subscriber Quota

The quota provisioning in SCA BB is prepared using subscriber quota buckets. Each subscriber has 16 buckets, and you can define each bucket for volume or sessions. When a subscriber uses a particular service, the amount of consumed volume or number of sessions is subtracted from one of the buckets. The service configuration, which is defined in the general policy definition by using the SCA BB Console, determines which bucket to use for each service. Consumption for the volume buckets is counted in units of L3 kilobytes and consumption for the session buckets is the number of sessions. For example, it is possible to define that the Browsing and E-mail services consume quota from Bucket number 1, P2P service consumes quota from Bucket number 2, and that all other services are not bound to any particular bucket.

Quota bucket comprises from the following components:

- Bucket ID—Unique identifier of the bucket **(String)** as defined in the predefined policy. Valid values are numbers in range [1-16]
- Bucket value—Quota bucket value **(long)**

Quota Operation dynamically modifies a subscriber's quota buckets. There are two types of quota operations:

- ADD_QUOTA_OPERATION—Adds the new quota value to the current value of the bucket residing on the SCE platform
- SET_QUOTA_OPERATION—Replaces the value of the quota bucket residing on the SCE platform with the new value

Examples

Current values of subscriber A's quota at the SCE are as follows:

Figure 4-1 Subscriber Quota - Current Values

We want to apply the following actions to the existing quota:

Figure 4-2 Subscriber Quota - Actions to Apply

After performing the quota actions, the result is:

Figure 4-3 Subscriber Quota - Results

For additional information about Subscriber Quota, see the *Cisco Service Control Application for Broadband User Guide*.

The following sections describe the classes the API provides for operations that include the subscriber quota management operations.

- SCAS_BB_Quota, page 4-6
- SCAS_BB_QuotaOperation, page 4-7

SCAS_BB_Quota

The SCAS_BB_Quota class implements the Quota interface, which the QuotaListenerEx interface uses in all callback functions. See Information About the QuotaListenerEx Interface Class, page 5-10.

The following method constructs the SCAS_BB_Quota based on the array of IDs and values:

```
public SCAS_BB_Quota (String[] bucketIDs,
                      long[] bucketValues)
```

The following method constructs the SCAS_BB_Quota based on the array of IDs and values, the profile ID, the reason, and the timestamp:

```
public SCAS_BB_Quota (String[] bucketIDs,
                      long[] bucketValues,
                      int quotaProfileId,
                      int reason,
                      long timestamp)
```

The following method allows retrieving of the quota buckets' IDs:

```
public String[] getBucketIDs()
```

The following method allows retrieving of the quota buckets' values:

```
public long[] getBucketValues()
```

The **quotaProfileId** parameter is the identifier for the quota profile, which is the package ID. The following method allows retrieving of the quota profile ID:

```
public int getQuotaProfileId()
```

The **reason** parameter is relevant only for quota status events and has three possible values:

- 0—The configured time was reached, for example, every two minutes
- 1—The quota status event was triggered by a subscriber logout
- 2—The quota status event was triggered by a package change

The following method allows retrieving of the reason:

```
public int getReason()
```

The **timestamp** parameter contains the time (in the SCE) at which the event was generated. It is calculated as the number of seconds from January 1, 1970 00:00 GMT.

The following method allows retrieving of the timestamp:

```
public long getTimestamp()
```

SCAS_BB_QuotaOperation

The SCAS_BB_QuotaOperation class implements the QuotaOperation interface, which is used for Subscriber Provisioning operations that include the subscriber's quota such as login operation (see Information About the login Operation, page 5-19) and update quota operation (see Information About the quotaUpdate Operation, page 5-28).

The following method constructs the SCAS_BB_QuotaOperation based on the array of IDs, values and actions:

```
public SCAS_BB_QuotaOperation (String[] IDs,
                               long[] values,
                               short[] actions)
```

The following method allows retrieving of the quota buckets' IDs:

```
public String[] getBucketIDs()
```

The following method allows retrieving of the quota buckets' values:

```
public long[] getBucketValues()
```

The following method allows retrieving of the quota buckets' actions:

```
public short[] getBucketActions()
```

Information About Bulk Operations Data Types

Use bulk classes and operations when performing the same method for many subscribers each with its own parameters. The API provides the bulk classes for result handling of bulk operations and for bulk indications from the SCE. The bulk classes are passed to the bulk methods such as **loginBulk** and **logoutBulk**.

The following is a list of considerations when using the bulk operations:

- All bulk classes are inherited from the common BulkBase class.
- Due to the memory constraints of the SCE, the bulk size is limited to a maximum of 100 entries.

Bulk Iterator

The BulkBase class provides an iterator to view the data contained in the bulk.

The following is the syntax for the Bulk Iterator:

```
Iterator getIterator()
```

This iterator can be used for iteration over the bulks received from the SCE in various indications (for example, logoutBulkIndication, loginPullBulkResponseIndication, and so forth) or for inspecting the data you provided to various operations in case an operation has failed.

The iterator provides the following methods for data retrieval:

```
public Object next()
public boolean hasNext()
```

The next() method returns a SubscriberData object.

The SubscriberData class is used for retrieving the information of a single subscriber contained within the bulk.

SubscriberData

The **SubscriberData** class represents all of the operations that can be performed on a specific subscriber. The SubscriberData class contains the following utility methods for information retrieval:

```
public String getSubscriberID()
public String getAnonymousID()
public String[] getMappings()
public short[] getTypes()
public boolean getAdditiveFlag()
```

The following sections describe various bulk data types that are available for different API operations.

Login_BULK Class

This class represents bulk of subscribers and it includes all data required for the **loginBulk** operation.

Constructor

To construct the **Login_BULK** filled with the data use the following constructor:

```
public Login_BULK(String[] subscriberIDs,
                  NetworkID[] networkIDs,
                  boolean[]networkIDsAdditive,
                  PolicyProfile[] policy,
                  QuotaOperation[] quota)
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

networkID—The network identifier of the subscriber. See Information About Network ID Mappings for more information.

networkIDAdditive—If this flag is set to TRUE, the supplied NetworkID is added to the existing networkIDs of the subscriber. Otherwise, the supplied networkID replaces the existing networkIDs.

policy—Policy profile of the subscriber. See Information About SCA BB Subscriber Policy Profile for more information.

quota—Quota of the subscriber. See Information About Subscriber Quota for more information.

To construct an empty Login_BULK, use the following method:

```
public Login_BULK()
```

addBulkEntry Method

Use the following method to add entries to the bulk:

```
public void addBulkEntry(String subscriberID,
                         NetworkID networkID,
                         boolean networkIdsAdditive,
                         PolicyProfile policy,
                         QuotaOperation quota)
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

networkID—The network identifier of the subscriber. See Information About Network ID Mappings for more information.

networkIDAdditive—If this flag is set to TRUE, the supplied NetworkID is added to the existing networkIDs of the subscriber. Otherwise, the supplied networkID replaces the existing networkIDs.

policy—Policy profile of the subscriber. See Information About SCA BB Subscriber Policy Profile for more information.

quota—Quota of the subscriber. See Information About Subscriber Quota for more information.

Examples

Login_BULK Object Usage: Example

This example demonstrates the usage of the Login_BULK object:

```
// Prepare all data for the bulk construction
String[] names = new String[5];
NetworkID[] mappings = new NetworkID[5];
boolean[] additive = new boolean[5];
PolicyProfile[] policy = new PolicyProfile[5];

for (int i=0; i<5; i++)
{
    names[i]="sub_"+i;
    mappings[i] = new NetworkID("1.1.1."+i,NetworkID.TYPE_IP);
    additive[i] = true;
    policy[i] = new PolicyProfile(new String[]{"packageId="+(i+1)});
}

// construct the bulk object
Login_BULK bulk = new Login_BULK(names,mappings,additive,policy,null);
// Now it can be used in loginBulk operation
sceApi.loginBulk(bulk,null);
```

Manipulating Login_BULK: Example

This example demonstrates an alternative way of manipulating the Login_BULK object:

```
// Construct the empty bulk
Login_BULK bulk = new Login_BULK ();

// Fill the bulk using addBulkEntry method:
for (int i=0; i<20; i++)
{
    String name ="sub_"+i;
    NetworkID mappings = new NetworkID(i+1);
    boolean additive = true;
    PolicyProfile policy = new PolicyProfile(new String[]{"packageId="+(i+1)});
    QuotaOperation quota = new SCAS_BB_QuotaOperation(
           new String[]{"1","2","3"},
           new long[]{80,80,0}
           new short[]{SCAS_BB_QuotaOperation.ADD_QUOTA_OPERATION,
                       SCAS_BB_QuotaOperation.ADD_QUOTA_OPERATION,
                       SCAS_BB_QuotaOperation.SET_QUOTA_OPERATION});
    bulk.addBulkEntry(name,mappings,additive,policy,quota);
}
// Now it can be used in loginBulk operation
sceApi.loginBulk(bulk,null);
```

SubscriberID_BULK Class

The **logoutBulkIndication** callback function that requires only subscriber IDs to be entered uses the SubscriberID_BULK class. See Information About the logoutBulkIndication Callback Method, page 5-10.

Constructors

To construct the **SubscriberID_BULK** with Subscriber IDs data, use the following constructor:

```
public SubscriberID_BULK(String[] subscriberIDs)
```

To construct an empty **SubscriberID_BULK**, use the following method:

```
public SubscriberID_BULK()
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

addBulkEntry Method

Use the following method to add entries to the SubscriberID bulk:

```
addBulkEntry(String subscriberID)
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

NetworkAndSubscriberID_BULK Class

Use the NetworkAndSubscriberID_BULK class in bulk operations that require Subscriber IDs and NetworkIDs in the following operations:

- **getSubscribersBulkResponse** callback (see the Information About the LoginPullListener Interface Class section)
- **logoutBulk** operation (see Information About the logoutBulk Operation)
- **networkIDUpdateBulk** operation (see Information About the networkIdUpdateBulk Operation)

Constructors

To construct the **NetworkAndSubscriberID_BULK** with the SubscriberID and NetworkID data, use the following constructor:

```
public NetworkAndSubscriberID_BULK(String[] subscriberIDs,
                                   NetworkID[] networkIDs,
                                   boolean[] netIdAdditive)
```

To construct an empty **NetworkAndSubscriberID_BULK**, use the following method:

```
public NetworkAndSubscriberID_BULK()
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

networkID—The network identifier of the subscriber. See Information About Network ID Mappings for more information.

networkIDAdditive—If this flag is set to TRUE, the supplied NetworkID is added to the existing networkIDs of the subscriber. Otherwise, the supplied networkID replaces the existing networkIDs.

addBulkEntry Method

Use the following method to add entries to the bulk:

```
addBulkEntry(String subscriberID,
             NetworkID networkID,
             boolean netIdAdditive)
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

networkID—The network identifier of the subscriber. See Information About Network ID Mappings for more information.

networkIDAdditive—If this flag is set to TRUE, the supplied NetworkID is added to the existing networkIDs of the subscriber. Otherwise, the supplied networkID replaces the existing networkIDs.

LoginPullResponse_BULK Class

This class represents a bulk of subscribers and includes all data required for the **loginPullResponseBulk** method.

Constructors

To construct the **LoginPullResponse_BULK** containing the relevant data, use the following constructor:

```
public LoginPullResponse_BULK(String[] anonymousSubscriberIDs,
                              String[] subscriberIDs,
                              NetworkID[] networkIDs,
                              boolean[] networkIdsAdditive,
                              PolicyProfile[] policy,
                              QuotaOperation[] quota)
```

To construct an empty **LoginPullResponse_BULK**, use the following method:

```
public LoginPullResponse_BULK()
```

Parameters

anonymousSubscriberID—The identifier of the anonymous subscriber. This is sent by the SCE within the loginPullRequest/loginPullBulkRequest indication (see Information About the loginPullRequest Callback Method and Information About the loginPullRequestBulk Callback Method). See Information About Subscriber Integration Models for more information.

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

networkID—The network identifier of the subscriber. See Information About Network ID Mappings for more information.

networkIDAdditive—If this flag is set to TRUE, the supplied NetworkID is added to the existing networkIDs of the subscriber. Otherwise, the supplied networkID replaces the existing networkIDs.

policy—Policy profile of the subscriber. See Information About SCA BB Subscriber Policy Profile for more information.

quota—Quota of the subscriber. See Information About Subscriber Quota for more information.

addBulkEntry Method

Use the following method to add entries to the bulk:

```
public addBulkEntry(String anonymousSubscriberID,
                    String subscriberID,
                    NetworkID networkID,
                    boolean networkIdAdditive,
                    PolicyProfile policy,
                    QuotaOperation quota)
```

Parameters

anonymousSubscriberID—The identifier of the anonymous subscriber. This is sent by the SCE within the loginPullRequest/loginPullBulkRequest indication (see Information About the loginPullRequest Callback Method and Information About the loginPullRequestBulk Callback Method). See Information About Subscriber Integration Models for more information.

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

networkID—The network identifier of the subscriber. See Information About Network ID Mappings for more information.

networkIDAdditive—If this flag is set to TRUE, the supplied NetworkID is added to the existing networkIDs of the subscriber. Otherwise, the supplied networkID replaces the existing networkIDs.

policy—Policy profile of the subscriber. See Information About SCA BB Subscriber Policy Profile for more information.

quota—Quota of the subscriber. See Information About Subscriber Quota for more information.

PolicyProfile_BULK Class

The **updatePolicyProfileBulk** operation uses this class that represents a bulk of subscriber IDs and subscriber policy profiles.

Constructors

To construct the **PolicyProfile_BULK** containing the relevant data, use the following constructor:

```
public PolicyProfile_BULK(String[] subscriberIDs, PolicyProfile[] policy)
```

To construct an empty **PolicyProfile_BULK**, use the following method:

```
public PolicyProfile_BULK()
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

policy—Policy profile of the subscriber. See Information About SCA BB Subscriber Policy Profile for more information.

addBulkEntry Method

Use the following method to add entries to the bulk:

```
public addBulkEntry(String subscriberID, PolicyProfile policy)
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

policy—Policy profile of the subscriber. See Information About SCA BB Subscriber Policy Profile for more information.

Quota_BULK Class

The following operations use this class that represents a bulk of subscribers IDs and subscriber quota buckets:

- getQuotaStatusBulk operation (only the bucket IDs are to be provided)
- quotaStatusBulkIndication callback method
- quotaDepletedBulkIndication callback method
- quotaBelowThresholdIndication callback method

Constructors

To construct the **Quota_BULK** containing the relevant data, use the following constructor:

```
public Quota_BULK(String[] subscriberIDs, Quota[] subscribersQuota)
```

To construct an empty Quota_BULK, use the following method:

```
public Quota_BULK()
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

quota—Quota of the subscriber. See Information About Subscriber Quota for more information.

addBulkEntry Method

Use the following method to add entries to the bulk:

```
public addBulkEntry(String subscriberID, Quota quota)
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

quota—Quota of the subscriber. See Information About Subscriber Quota for more information.

QuotaOperation_BULK Class

The **QuotaUpdateBulk** operation and the login operation use this class that represents a bulk of subscribers IDs and subscriber Quota operations.

Constructors

To construct the **QuotaOperation_BULK** containing the relevant data, use the following constructor:

```
public QuotaOperation_BULK(String[] subscriberIDs,
                           QuotaOperation[]quotaOperations)
```

To construct an empty QuotaOperation_BULK, use the following method:

```
public QuotaOperation_BULK()
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

quotaOperation—The quota operation to perform on the quota of the subscriber. See Information About Subscriber Quota for more information.

addBulkEntry Method

Use the following method to add entries to the bulk:

```
addBulkEntry(String subscriberID, QuotaOperation quotaOperation)
```

Parameters

subscriberID—The unique ID of the subscriber. See the Subscriber ID section for the subscriber ID format description.

quotaOperation—The quota operation to perform on the quota of the subscriber. See Information About Subscriber Quota for more information.