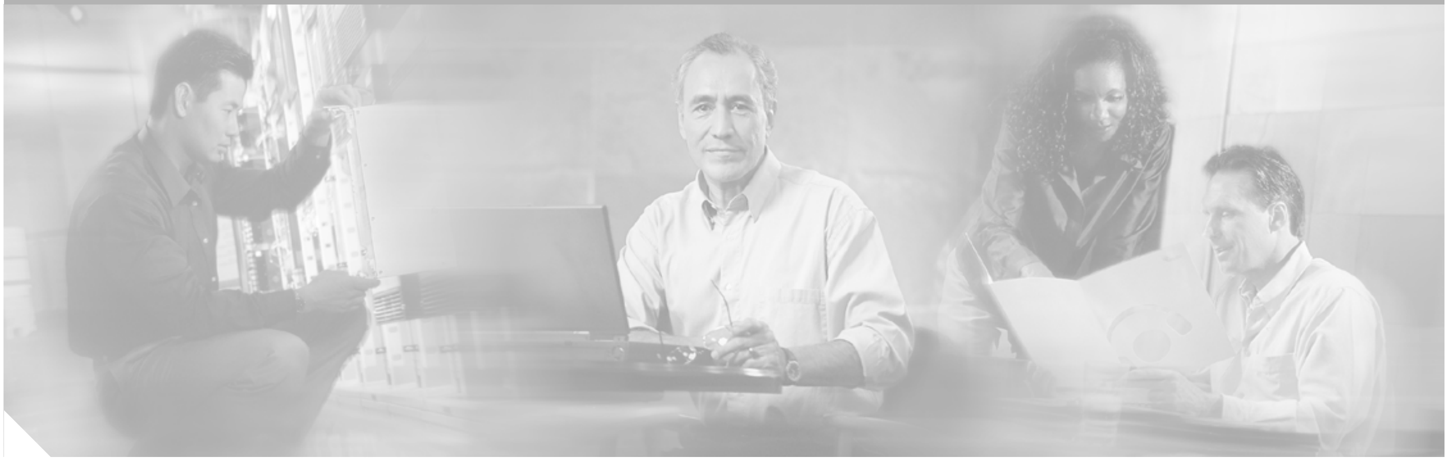


Cisco Capacity Admission Control Manager User Guide

Software Release 1.6
December 22, 2006

Corporate Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 526-4100

Customer Order Number:
Text Part Number:

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB’s public domain version of the UNIX operating system. All rights reserved. Copyright 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Cisco Capacity Admission Control Manager User Guide

Contents

Preface

Overview

This document discusses the Admission Control Manager offering. It discusses its logical and deployment architecture.

Audience

This guide is for the network professional who handles the CAC module.

Organization

This document contains five chapters and three appendices:

- Chapter 5 - Introduction
- Chapter 4 - Architecture
- Chapter 4 - Deployment
- Chapter 4 - Topology
- Chapter 5 - Service Control Engine
- Appendix A - Glossary
- Appendix B - SCE Error Codes
- Appendix C - API Classes

Documentation Set

The documentation for your Broadband Policy Manager (BPM) system includes the following documents:

- Cisco Broadband IP Service Module User Guide
- Cisco Broadband Policy Design Studio User Guide
- Cisco Broadband Policy Manager Installation and Configuration Guide
- Cisco Broadband Policy Manager Operations Guide
- Cisco Broadband Policy Manager Release Notes
- Cisco Capacity Admission Control Manager User Guide

Cisco Broadband IP Service Module User Guide

This document discusses the Broadband IP Service Module for session management and network adaptation. It discusses its architecture, components, access methods, and functions.

Cisco Broadband Policy Design Studio User Guide

This guide provides instructions for installing the Broadband Policy Design Studio (BPDS). It discusses how to use the BPDS to create, deploy, and manage network services and topologies.

Cisco Broadband Policy Manager Installation and Configuration Guide

This guide describes how to install the software for the BPM. It describes how to install and configure the Solaris operating system for use by the BPM. It also includes procedures to install and configure the BPM software and the procedures to install and log into the BPDS.

Cisco Broadband Policy Manager Operations Guide

This guide describes the use of the BPDS to obtain information, conduct day-to-day operations, perform maintenance tasks, and troubleshoot problems with the BPM system. These tasks include use of the Log Messages addendum, the Application Log Messages addendum, and the Statistics addendum.

Cisco Broadband Policy Manager Release Notes

This document describes new features, known limitations, and other important information about the BPM system.

Cisco Capacity Admission Control Manager User Guide

This document discusses the architecture and components for the Capacity Admission Control Manager product.

5

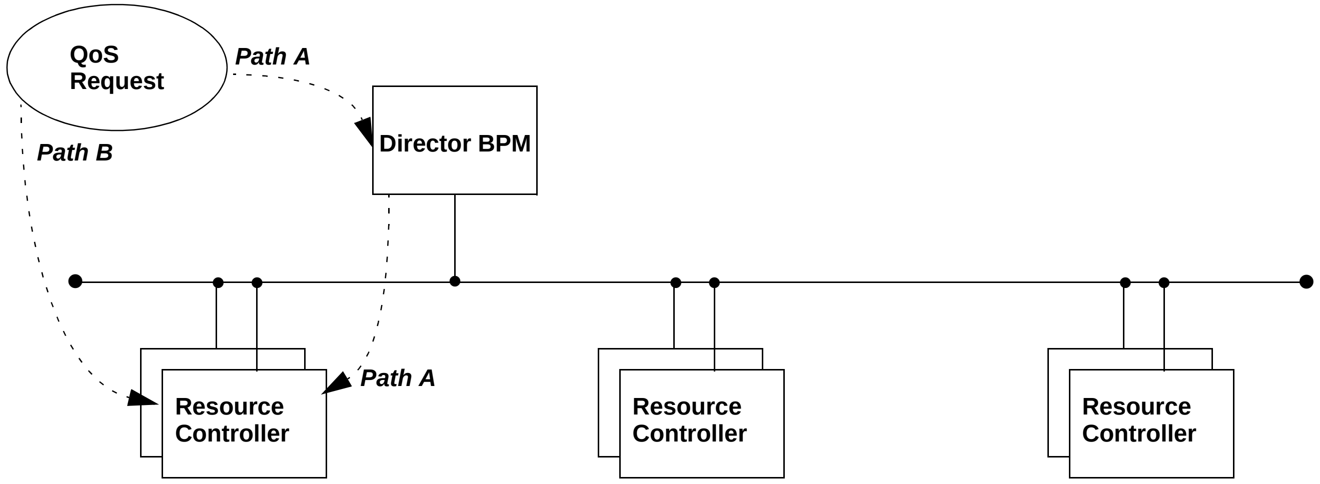
Introduction

Overview

The Admission Control Manager (ACM) product runs on the Resource Controller Broadband Policy Manager (BPM) and the BPM Director. The ACM product provides Capacity Admission Control (CAC) services over broadband access networks. Admission control monitors, controls, and enforces the use of network resources and services with policy-based management. The criteria for policy-based management includes user and application identification, or traffic identification, based on how, when, and where it enters the network.

Depending on the application in use, resource requests can take diverse paths to reach the Resource Controller, as depicted in Figure 1.

Figure 1. Example: Request Paths to Resource Controller



In a bandwidth-managed network with Quality of Service (QoS) features, sessions compete for the available network bandwidth.The ACM product estimates the QoS level required for a new user session and determines if enough bandwidth is available. If sufficient bandwidth is available, it allows the session onto the network.

BPM

The Broadband Policy Manager (BPM) consists of visual development, deployment manager, and execution environments that provide an architecture for service-oriented systems. The BPM simplifies the repeated production of related solutions for real-time network policy management.

BPDS

In the BPM architecture, dataflow programming promotes the data movement and transformation in program execution. The Broadband Policy Design Studio (BPDS) graphical user interface (GUI) facilitates the tasks of the service designer and network administrator. Using the BPDS, the service designer develops programs interactively using drag-and-drop visual programming. The programs move data between operators that are exposed by software agents. The agents encapsulate specific implementations protocols, network devices, data sources, or logic capabilities. The designer can combine the programs into collections of services, referred to as applications, which provide complete solutions. The network administrator publishes these applications in the execution environment.

Execution Environment

The execution environment is a distributed domain of networked processing nodes, links, and resources. A node is a computer or some other device on a network with a unique network address. A link is a line or channel between the nodes, over which data is transmitted. Information resources can be attached to a node to describe its capabilities, for example, its Quality of Service attributes. Resources can also be attached to a link to describe, for example, its bandwidth capabilities or delay properties.

Each node runs a highly concurrent graph-traversal engine, coupled with a fast data switching fabric. Once published to an execution environment on a node, application services are available for execution and monitoring. The execution environment also provides resiliency and failover capabilities.

System Administration

Service providers use the BPM product suite to create and deploy advanced services on broadband networks. The service designer develops the programs or services. The network administrator performs routine operational, maintenance, and troubleshooting tasks, including starting or stopping the BPM, monitoring status, obtaining statistics or other information, managing user access, configuring components, solving problems, and deploying services.

The administrator has various tools available to perform these tasks. These include the BPDS and a command line interface. The BPDS includes the following views:

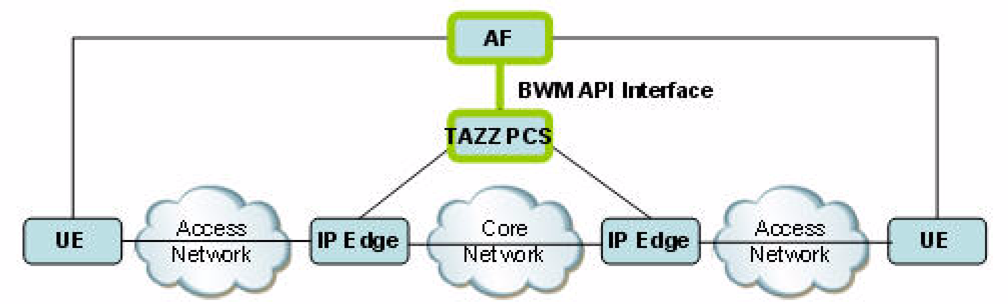
- Network Administration screen, which shows BPM data from a network point of view, including service engines, agents, and services
- Subscriber Access screen, which shows BPM data from a subscriber access point of view, including access groups, access lines, administrators, network devices, policies, servers, and subscribers
- Service Design screen, which shows the structure of existing services and allows the creation of new services.

In addition, the administrator can use the command line interface (CLI) to directly enter certain commands.

Application Interface

The Bandwidth Manager (BWM) Application Programming Interface (API) provides a proprietary, open, interface that permits integration with the BPM. The API allows an application to request microflow reservations between IP endpoints on an end-to-end IP network. Figure 2 presents the API interface.

Figure 2. BWM API Interface



The BWM API supports the following admission control use cases:

- Single-ended Access Reservation
- Single-ended Access and Core Reservation
- End-to-End Access Reservation
- End-to-End Access and Core Reservation

The API allows external systems to add, update, and remove nodes, links, and resources from the topology model. The API provides facilities for integration with carrier Operation and System and Support (OSS).

- The system exposes a Web Services API that allows manipulation of individual topology elements.
- The system exposes a Web Services API for bulk provisioning.
- The system supports SOAP over HTTP.
- The system supports SOAP over JMS.
- The system exposes an SNMPv2 interface that allows manipulation of topology data.

The ACM product supports the following Access Provider networks types:

- Digital Subscriber Line
- Asynchronous Transfer Mode
- Ethernet

See Chapter 4, Topology, for a further discussion of these network types.

Realms

Topology, resources, active sessions, and active contexts exist in an information *realm*. The realm improves performance by restricting lookups and updates against smaller data sets, providing less lock contention and faster search times. It also allows a Resource Controller to comprehend the realms for which it is responsible. See Chapter 4 for a further discussion of Realms.

Network Topologies

The ACM product supports the following Access Provider networks types:

- Digital Subscriber Line
- Asynchronous Transfer Mode
- Ethernet

See Chapter 4 for a discussion of the network topologies, bandwidth considerations, and discovery and synchronization.

Additional Features

Table 1 presents additional control features enabled by the ACM product that are not necessarily involved with CAC. These features can be used on a standalone basis or as part of a larger solution when combined with the ACM product.

Table 1. Additional Features.

Service	Description
Bandwidth on Demand	Subscriber can dynamically select from different access speeds for his or her network connectivity.
QoS on Demand	Subscribers and applications can dynamically allocate QoS queuing facilities at service, subscriber, and session levels.
Dynamic Charging	Applications can dynamically control online and offline charging rules enforced at service, subscriber, and session levels.
Subscriber Self-Service	Subscriber can switch between different access products in real-time. The product offerings combine other services in this table.
Parental Control	Traffic filters prevent access to objectionable or fee-based content on the network. Filter rules for this feature can change frequently.
Walled Garden	Subscribers can access a subset of network resources that can include general Internet access. Examples: public access WLAN and sponsored services.

4

Architecture

Overview

This chapter discusses the Capacity Admission Control CAC components.

CAC Modules

The CAC components use the following architectural modules. Each module is embodied as an agent and/or a set of flows.

- Topology Store Function
- Topology Awareness Function
- Session Awareness Function
- Context Timer Function
- Accounting Log Function
- Statistics Function
- Alarm Notification Function

Topology Store Function

The Topology Store Function (TSF) maintains the Topology Information Model (TIM) for a given Resource Controller system component. Topology updates are received from the Topology Database Server (TDS), via the component Topology Awareness Function (TAF). The TSF stores those updates in the TIM. The TIM is stored in a database as a group of nodes connected by links. The nodes represent physical or abstract network devices. The links represent physical or abstract connectivity between the nodes. Information (resources) can be *attached* to nodes to describe their capabilities (for example, IP address maps). Information (resources) can also be *attached* to links (for example, bandwidth capabilities, delay properties, service function). The TSF supports the concept of distinct information realms, allowing the application to segment topology and resource representation for increased performance and stability. Each node has a unique identifier, which can be dependent on the deployment scenario. Each link is stored as a directed association between two nodes.

Topology Awareness Function

The Topology Awareness Function (TAF) extracts and reacts to changes in the underlying network. This information can be read from provisioning files, or it can be received from the Topology Database Server. The information is stored via the Topology Store Function, which maintains the information model in a data store.

Session Awareness Function

The Session Awareness Function (SAF): The SAF obtains dynamic session information. This relates to individual subscriber sessions (IP), subscriber network sessions (such as PPP), or possibly higher-level session concepts such as IP pools. Each of these concepts represents a separate SAF.

Context Timer Function

The Context Timer Function (CTF) accomplishes soft-state semantics for contexts and the resources they consume by periodically releasing expired contexts and recouping their resources. It does this by interfacing with the Admission Control Function (ACF), so it can perform QoS release capability and with the TSF to retrieve and remove contexts.

Accounting Log Function

The Accounting Log Function (ALF) records entrance parameters, internal decisions, and exit responses, for business logic and handler flows. The invoker provides accounting-pertinent information and a correlation identifier. The ALF appropriately stores the information.

Statistics Function

The Statistics Function (SF) records and queries system statistics. It provides a location for components to store runtime state statistics for inspection by other clients.

Alarm Notification Function

The Alarm Notification Function (ANF) alerts external systems of aberrant behavior in the BPM by issuing SNMP traps. The ANF allows various components of the BPM to report unexpected conditions and behavior, as well as unexpected life cycle changes in a controlled and consistent manner.

4

Deployment

Overview

The CAC deployment consists of the following Broadband Policy Manager (BPM) systems:

- Director
- Resource Controller
- Network Topologies
- Domain Controller

This chapter discusses the deployment architecture.

Director

The Director provides an abstracted view of the entire deployed system towards the application layer. It handles the message routing functions between the applications and the network, Resource Controller, layer. All Directors share the same real-time mapping of sessions to Resource Controllers. To accomplish the Capacity Admission Control (CAC) goals, the Director BPM requires the following from the Topology Information Model (TIM):

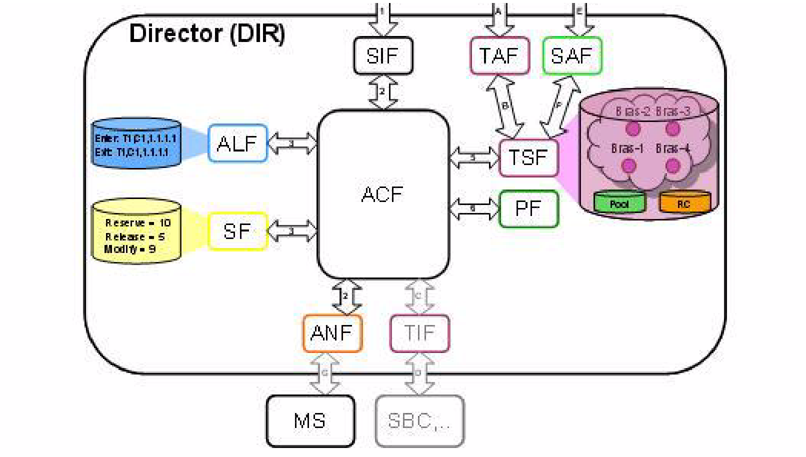
- IP address pool assignments to each Broadband Remote Access Server (BRAS)
- Resource Controller BPM component (active and standby address) assigned to each BRAS

A Director forwards a Quality of Service (QoS) request to the correct Resource Controller by mapping the IP addresses involved in the request to a BRAS and mapping the BRAS to the appropriate Resource Controller. The Director maintains all mappings in the Director Realm database tables. The Topology Database Server maintains the Director Realm. The Topology Database Server automatically distributes Director Realm information to a Director when the Director starts and when the Director Realm content changes.

The Director Realm contains a node for each BRAS. Resources representing IP Pools are attached to the nodes. The IP Pools identify ranges of IP addresses handled by each BRAS. Also, additional resources attached to each BRAS node in the Director Realm identify the active and standby Resource Controllers that handle the BRAS. A Director uses the IP Pool resources and the Resource Controller resources of the Director Realm to forward an incoming call request to the correct Resource Controller

Figure 3 presents the internal structure of the Director.

Figure 3. Director Internal Structure



The Director encapsulates a specific type of Admission Control Function (ACF) that uses the Address Pool Information to determine the responsible Resource Controller components. It augments the request with local context (the path segment for which the Resource Controller is responsible). The Director forwards it to the Resource Controller components. The ACF collects the responses and aggregates the results into a unified response for the QoS request. For differing responses from the Resource Controller, the ACF restores appropriate state to the Resource Controller components.

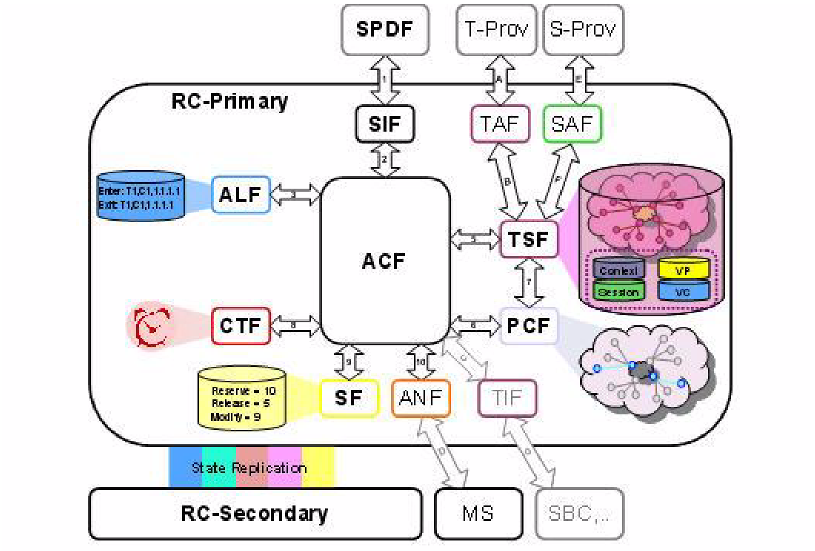
The Director represents:

- Signaling Interface Function (SIF), to interface with the external application function.
- Admission Control Function (ACF), to perform a unified admission control decision.
- Topology Store Function (TSF), to store the Director TIM, which holds pool and Resource Controller entries for each Aggregating Access Node (BRAS).
- Topology Awareness Function (TAF), to react to external updates to the TIM and filter them into the internal TSF.
- Accounting Log Function (ALF), to record accounting information.
- Statistics Function (SF), to record statistics.
- Alarm Notification Function (ANF), for alarm notification.

Resource Controller

The Resource Controller tracks resource utilization for the system. Resource Controllers can be configured in resilient pairs. The system of a given pair that handles requests is the *active* system. Its backup system is the *standby.* Each Resource Controller (or resilient pair) tracks the resources of a subset of the network topology, including logical and/or physical network entities, such as ATM Virtual Paths (VP) and/or Virtual Circuits (VC).

Figure 4. Resource Controller Internal Structure



The Resource Controller encapsulates an ACF that performs the access network-level admission control decision, based on resource utilization (Figure 4). The Resource Controller represents:

- Signaling Interface Function (SIF) to interface with the external application function, which is the Director. Since this is internal, the SIF is transparent.
- Topology Store Function (TSF) to maintain the TIM that the Resource Controller is responsible for and that resource consumption is checked against
- Topology Awareness Function (TAF) to interact with the TSF to react to topology changes (link or node activate/deactivate) by correctly cleaning up internal state and resource utilization
- Path Computation Function (PCF) to determine the path through the local topology that the call transits
- Context Timer Function (CTF) to enact the soft-state reservation model to automatically remove orphaned or stale contexts and the resources they consume
- Admission Control Function (ACF) to perform local admission control decision, based on TIM resource utilization
- Accounting Log Function (ALF) for recording accounting information
- Statistics Function (SF) for recording statistics
- Alarm Notification Function (ANF) for alarm notification

The Cisco solution provides a fully compliant Resource and Admission Control Subsystem (RACS) implementation and can extend support to non-RACS compliant policy enforcement points and complex service delivery networks that are not supported by the RACS architecture (for example, Deep Packet Inspection devices, MPLS PE routers, and DSLAMs).

At the Resource Controller, resource layers for the network, such as VP and VC priority queues (PQs), track resources. The deployed product is (or can be configured to be) cognizant of the resource types. Additionally, the Edge Admission Control (EAC) must understand the contexts existing over any link in its topology. The system uses this information to clean up resources appropriately if a link fails, or when a (PPP) session ends. The EAC accomplishes this by automatically creating a resource type called *CONTEXTS* and assigning one to each link. This resource tracks all active contexts over a link on the local topology.

Resource Controllers are usually dedicated to specific network resources (devices, links, etc.). The Resource Controller Layer is responsible for dynamically executing the fine-grained configuration changes to the underlying network devices and tracking the availability of the network resources under their control. A Resource Controller receives messages from the Director Layer indicating the policy actions required to fulfill the application or subscriber request.

Session Information Model

The Session Information Model (SIM) is internal to the TIM. It has a dedicated role and connects dynamic sessions with the more static topology. For DHCP-derived addresses, it dynamically associates an IP address with a CPE device. The Resource Controller uses the dynamic mapping of the IP address to the session to determine the CPE and BRAS devices associated with a particular IP address when it is in use. This determines the topology elements terminating the context, so that the PCF can compute a path between the elements in the local topology.

Directors are unconcerned with the dynamic nature of each IP address as assigned for each session. They resolve an IP address to a particular BRAS and the Resource Controller responsible for that BRAS. They do this with a best *first match* against all assigned IP address pools, each of which is assigned to a particular BRAS. Each BRAS has an assigned Resource Controller. The Director augments the request with the BRAS information for each side of the call. This information determines the *realm* for the Resource Controller. The Director forwards the request to the determined Resource Controller component, when more localized information is present.

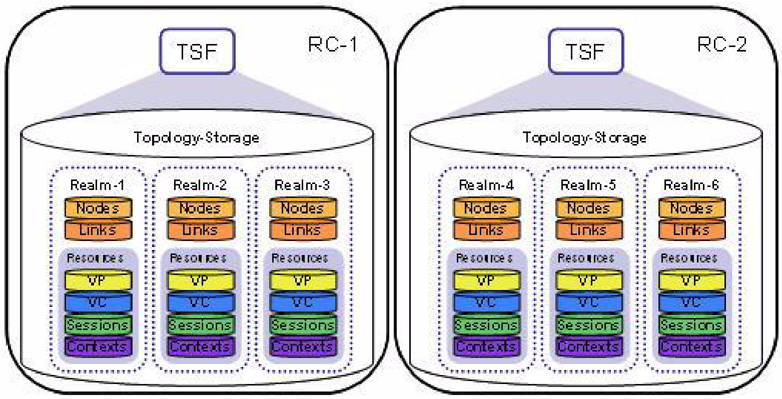
Realms

Topology, resources, active sessions, and active contexts exist in an information *realm*. The realm improves performance by restricting lookups and updates against smaller data sets, providing less lock contention and faster search times. It also allows a Resource Controller to comprehend the realms for which it is responsible. If a request involves a realm that the Resource Controller does not own, it may ignore the request (if that is the configured behavior). When introducing new Resource Controllers, BRAS responsibility is migrated from one Resource Controller to a new Resource Controller. The realm concept allows the information model to consider this a block movement of a realm. The migration affects only the realm that is moving. When scaling the BPM to accommodate more hardware and repartitioning the realms, only the realm/BRAS being moved is unavailable to QoS requests. Calls originating or terminating in other realms remain uninterrupted.

A BRAS defines a realm of self-contained information. The mapping of Resource Controllers to BRASs allows multiple BRASs and their access legs to be handled by a single Resource Controller. The state maintenance of various components is specified at the granularity of the Resource Controller. Thus, a single Topology Store Function (TSF) element handles more than one BRAS.

Figure 5 depicts the realms on a Resource Controller.

Figure 5. Realms on Resource Controllers



Each BRAS and its access legs are completely separate realms. No topology information is shared between the BRASs. An active session is active on only one BRAS. A context (call) can span more than one BRAS. A point-to-point call involves, at most, two BRAS nodes. The sessions and contexts are intrinsic resources for the QoS application. Virtual Paths (VP) and Virtual Connections (VCs) are used for a particular type of admission control.

Domain Realm

The Domain Realm maintains application level information about the physical network topology. The nodes in the topology represent Director and Resource Controller systems. The Topology Database Server uses the Domain Realm to understand the system topology. Links represent connectivity between cluster pairs. Resources represent interfaces on the component systems, system health, cluster information, and system configuration

Director Realm

The Director Realm stores information required by Director systems, including information about network devices (such as BRAS devices). The information specifies the Resource Controller responsible for each device and the IP address pools each device handles. A Director uses this information to forward an incoming request to the correct Resource Controller. The Topology Database Server maintains the Director Realm, and the server distributes its updates to each Director when updates occur.

Resource Realm

A Resource Realm represents a BRAS device and its connected CPE equipment. The Resource Realm is provisioned on the Topology Database Server and distributed to the Resource Controller that coordinates activity for that BRAs. At runtime, the Resource Realm stores capacity and usage information required to perform CAC decisions

Topology Database Server

The Topology Database Server utilizes industry-standard database technology that allows any of the underlying system elements to interrogate it. The Topology Database Server provides external interfaces that enable customer provisioning and profile storage functions to populate the database. It maintains the Director Realm and automatically distributes Director Realm information to a Director when the Director starts and when the Director Realm content changes.

Domain Controller

The Domain Controller BPM is a standalone system responsible for total domain management, including application deployment, configuration, and health for all systems in the domain. Each BPM system in the domain contains specific configured sections of the total deployment. The entire system configuration must be consistent for proper operation. The Domain Controller maintains the consistency of the component configurations in the entire overall solution.

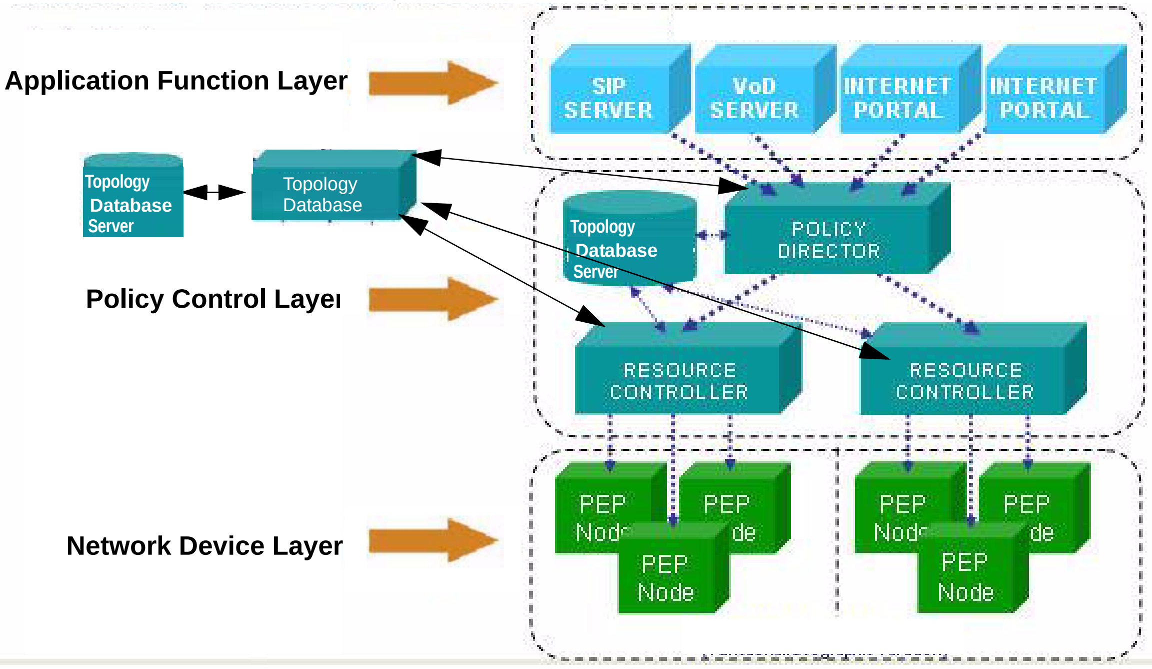
Only one Domain Controller exists per domain. It manages the process for its domain and configures everything beneath it. It initiates configurations, system packages, and tzz packages. It administers element configuration controls for operations such as resiliency. The Domain Controller communicates with other systems via the TRPC protocol.

The Domain Controller provides a control point through which the administrator can manage the entire deployment. The administrator connects to the Domain Controller via the Broadband Policy Design Studio (BPDS), the graphical user interface, to manage all members of the domain.

Layout

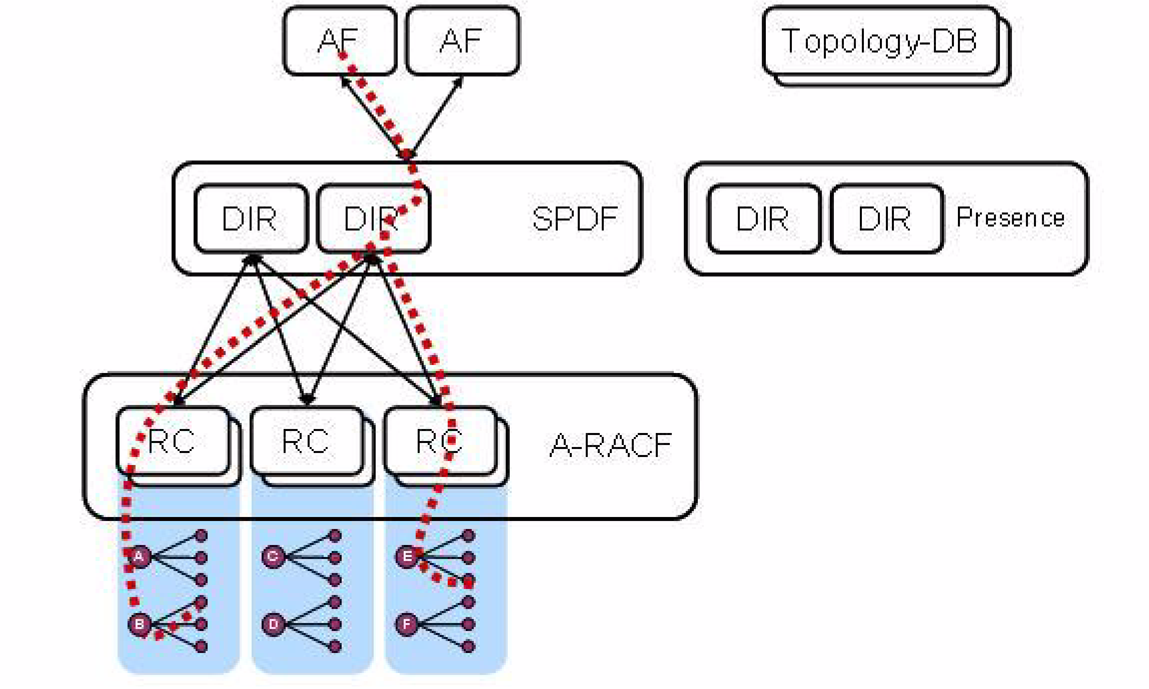
The requesting applications are received at the Application Function Layer and abstracted from the network by the Policy Control Layer. The Policy Control Layer handles the northbound abstraction, hybrid southbound control, and the implementation of business rules. Figure 6 presents a general view of the architecture.

Figure 6. Technology Deployment



The functional layout of machines and roles for the EAC primarily involves Directors and Resource Controllers (Figure 7). Each Resource Controller is responsible for a portion of the access network. Resource Controllers are deployed redundantly, with a standby Resource Controller for every active Resource Controller. The Director determines the access networks in use, queries the appropriate Resource Controllers, and determines the unified result. The Topology-DB houses the global TIM. Changes to this information base are infrequent, but the information is resilient.

Figure 7. Deployment Layout



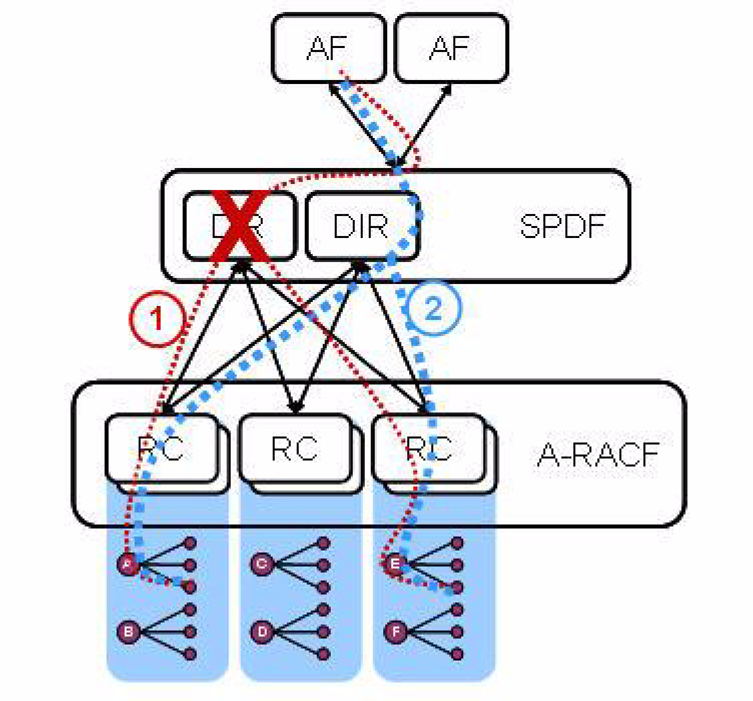
Resiliency

Capacity Admission Control delivers a highly available capacity reservation service by employing a variety of resiliency mechanisms in various capacity admission control functional areas. The Directors are stateless, and can achieve high availability by being deployed in an N + 1 active/active manner. An essential part of the capacity admission control solution design is load balancing across the N+1 set of Directors.

The Resource Controllers handle a specific access network section, and are highly stateful. Resource Controllers are paired in a 1:1 active/standby manner. Each Director is responsible for an access network section and switches to the standby Resource Controller if the active Resource Controller does not respond. Other elements, such as the Topology Database Server, provide independent resiliency.

Failure of a Director does not cause a lasting problem for system performance or availability. Directors are stateless and can be deployed in N + 1 active/active configuration. They can be load-balanced. This allows for adequate availability and performance scaling characteristics by adding more Directors (Figure 8).

Figure 8. Director Failure



When a failed Director restarts, it synchronizes its TIM with the topology database. It re-establishes connections with the Resource Controllers. It is ready to receive requests from AFs. Director\s must know the state of active and standby Resource Controllers responsible for access networks. The Director detects delayed response times or dropped requests and notifies the Topology Database Server. The Topology Database Server initiates Resource Controller failover, if required.

4

Topology

Overview

The ACM product supports the following Access Provider networks:

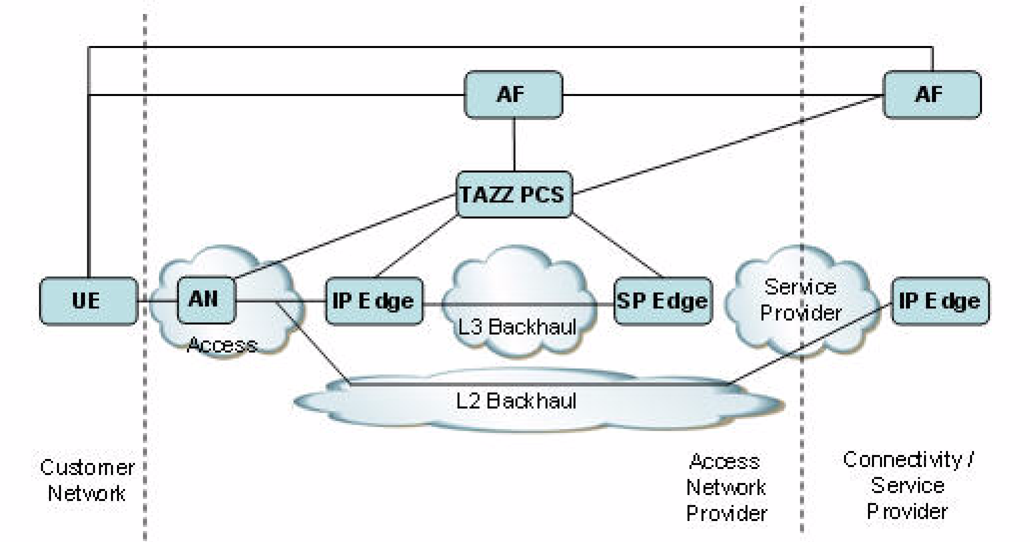
- Digital Subscriber Line
- Asynchronous Transfer Mode
- Ethernet

This section presents their topologies, bandwidth considerations, and network discovery and synchronization.

Digital Subscriber Line

The Broadband Policy Manager (BPM) supports the Digital Subscriber Line (DSL) Access Network architectures documented in the DSL Forum Documents TR-059 and TR-101. Figure 9 depicts the network architecture supporting the DSL topology information model.

Figure 9. DSL Access Network Architecture



This network architecture supports the topology information model in Figure 10, which depicts the DSL AN Access Realm and DSL AN Service Realm.

Figure 10. Supported Topology Information Model

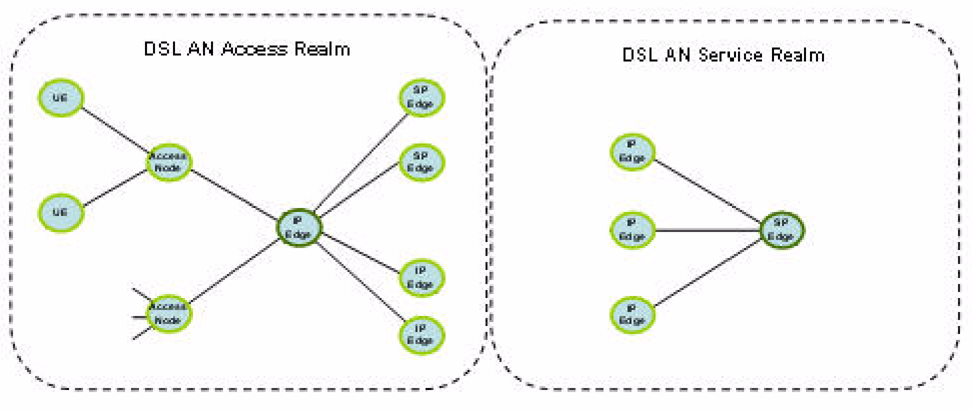
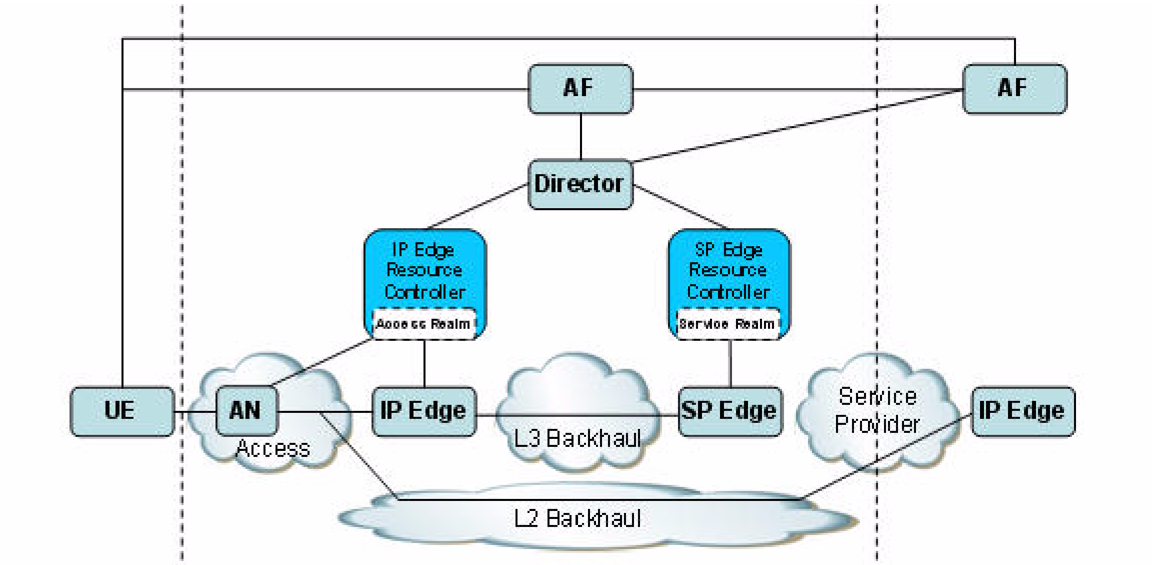


Figure 11 depicts the DSL AN Access Realm and DSL AN Service Realm, which are deployed in IP Edge and SP Edge Resource Controllers. The Resource Controller BPM tracks resource utilization for the system. Each Resource Controller tracks the resources of a subset of the network topology.

Figure 11. DSL Access Node Realm Model



Asynchronous Transfer Mode

The Asynchronous Transfer Mode (ATM) model extends the basic DSL access model in the following ways:

- The IP Edge to AN links are Virtual Paths (VPs), identified by Virtual Path Identifiers (VPIs).
- The AN to UE links are Virtual Channels (VCs), identified by a Virtual Channel Identifiers (VCIs).
- A UE can be associated with only one IP Edge device.
- VPs and VCs have associated traffic descriptors.

Ethernet

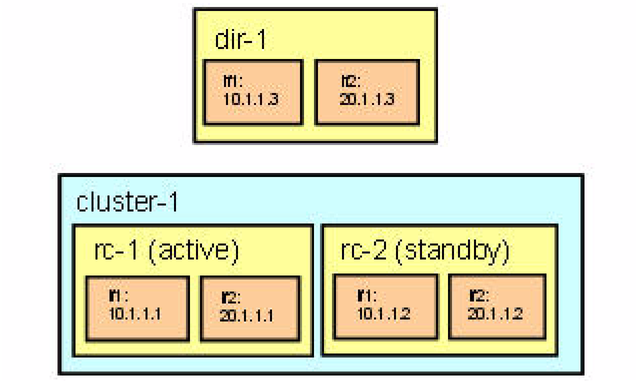
The Ethernet model extends the basic DSL access model in the following ways:

- IP Edge to AN links are as S-VLANs, identified by an outer VLAN tags.
- AN to UE links are C-VLANs, identified by VPIs.
- A UE may be associated with multiple IP Edge devices.
- S- and C-VLANs have associated traffic descriptors.

Resource Controller Realm Topology

A Resource Controller realm record defines the topology information required by Resource Controllers. Figure 12 presents a Topology Information File.

Figure 12. Example Topology Information File



A Resource Controller realm directive appears below.

The "<id>" and "<action>" fields are defined earlier in this document.

<realm id='<id>' type='tsf:realm' action='<action>'>

where,

- ID: The unique ID of the realm. In the case of a Resource Controller realm, this identifies a BRAS device.
- Action:
  - **add**: The Realm is added to the topology. Errors are generated for any data that already exists in the topology (however processing of the file continues after an error is encountered.)
  - **remove**: The realm is removed from the topology.
  - **update**: The realm should already exist in the topology, and is updated per the data in this file.
  - (empty string): No action is taken for the realm itself. Component resources of the realm may be added/removed/updated, as specified by the content of this directive.
  - <Unspecified>: If *action* is not specified, the default action as specified in the <TIM> directive, or in the tash **Provision** command used to load this file, is assumed.

Within A Resource Controller realm record, the following subrecords are allowed.

- <resource>: This defines a resource of the topology, used to represent VP capacity and VC capacity. The <resource> directive appears as follows.

  <resource id='<id>' type='<type>'>

where

- id: is the ID of the resource within the realm. The value depends on the type of the resource. For VPs, this form is "<bras>-<port>-<vp>-us" for upstream capacity, and "<bras>-<port>-<vp>-ds" for downstream capacity.
- For VCs, this form is "<bras>-<port>-<vp>-<vc>-us" for upstream capacity, and "<bras>-<port>-<vp>-<vc>-ds" for downstream capacity. In each of these, "<bras>" represents the ID of the BRAS node in the topology. "<port>", "vp", and "vc" are integers.
- type: This is one of the following values:

  - vp (for a resource used for VP capacity)
  - vc (for a resource used for VC capacity)

Within a resource record, the following subrecords are allowed.

- Property: Represents a property of a resource. The <property> directive appears as follows.

  <property name='<name>' value='<value>'>

Table 2 presents the name/value pairs for a domain realm property.

Table 2. Domain Name Properties

Resource Type	Name	Definition	Property Value
vp or vc	used	This is always zero on the TDS. It is set by a running Resource Controller during normal operation.	This is zero on a TDS. On a running Resource Controller, it represents the in-use capacity.
vp or vc	capacity	The capacity of the VP or VC	This is an integer, representing the capacity. No units are defined. The units are whatever the requesting interface requires, for example bits per second or Kbits per second. The administrator must provision this with correct values.

- <node>: Identifies a node within the topology being described. In a Resource Controller realm, the nodes represent the BRAS and the connected customer premises equipment devices. The <node> directive appears as follows.

  <node id='<id>' active='<flag>'>

where,

- <id>: The ID of a given node.
  - For a BRAS, this form is "rA_B_C_D", where "A", "B", "C", and "D" are the components of the BRAS's IP address "A.B.C.D"
  - For a CPE device, the node ID form is "<BRAS-ID>-<port>-<vp>-<vc>" where "<BRAS-ID>" is the ID of the BRAS for the realm, and "<port>", "<vp>", and "<vc>" represent how the BRAS is connected to the CPE device.
- <active>: A string that determines if the given node, or the link between nodes is up ("true") or down ("false").
- <link>: Identifies a link within the topology being described. In a Resource Controller realm, the links represent the connectivity between the BRAS and CPE devices. The BRAS is connected to each CPE devices by two links, one for upstream bandwidth and one for downstream bandwidth. The <link> directive appears as follows.

  <link id='<id>' src='<source>' dst='<dest>' active='<flag>'>

where,

- <id>: The ID of a given link. Within a Resource Controller realm, this takes the form "<source>=<dest>" where both "<source>" and "<dest>" represent either the node ID of the BRAS or of one of the CPE devices, depending on whether the link represents an upstream or downstream capacity, for example, "r192_9_200_15=r192_9_200_15-0-1-1".
- <source>: The source of a link. This represents the source of the link - the ID of either the BRAS or a CPE device, for example, "r192_9_200_15".
- <dest>: The destination of a link. This represents the source of the link - the ID of either the BRAS or a CPE device, for example, "r192_9_200_15-2-1-2".
- <active>: A string that identifies if the given cluster link is active. "true" or "false".

Within a link record, the following subrecords are allowed.

- <resourceid>: Represents a resource associated with the link. The <resourceid> directive appears as follows.

  <resourceid id=<id>' type='<type>'>

Table 3 presents the type/ID pairs for a domain realm *resourceid*.

Table 3. Domain Realm resourceid Type/ID Pairs

Type	ID	Definition
vp	A string, such as "r192_9_200_15-3-4-ds".	The ID of a VP resource.
vc	A string, such as "r192_9_200_15-3-4-2-ds".	The ID of a VC resource.

Example Topology

Figure 12, Figure 13, and Figure 14 define a topology consisting of a Domain Controller, one Director, and a clustered pair of Resource Controllers. The Resource Controllers serve one BRAS device, which contains two ports. Each port has two VPs, and each VP has two VCs.

Figure 13. Director Realm Structure

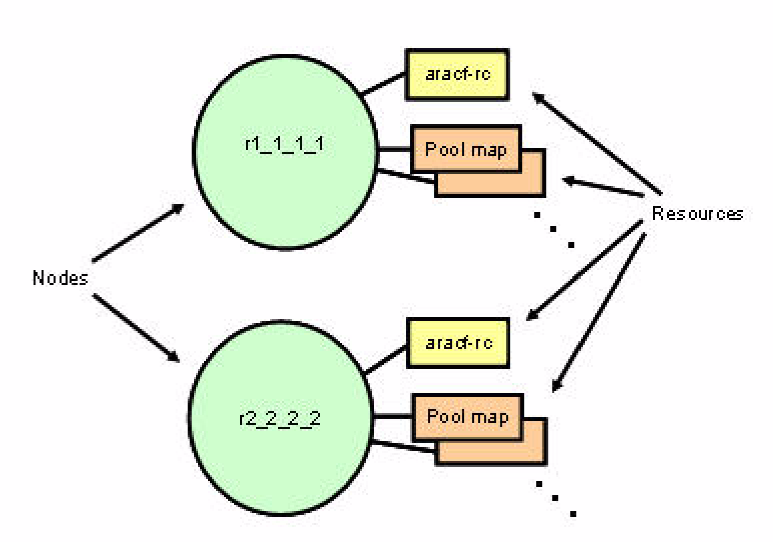
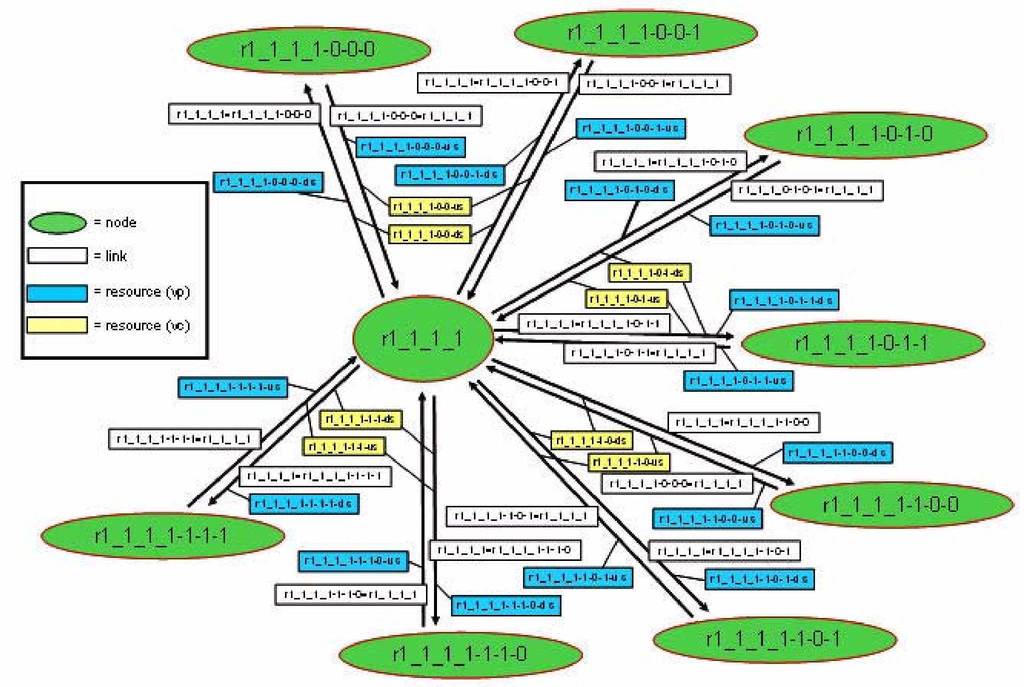


Figure 14. Sample Resource Controller Realm



Sample Topology Information File

```
<global>

  <!-- Domain, (taf_ services only affect local store) -->
  <tim realms='taf_realms' resources='taf_resources'
   nodes='taf_nodes' links='taf_links'>
    <!-- example domain information: 3 systems -->
    <!-- 2 RC's (.1, .2) that are clustered, and 1 dir (.3) -->

    <!-- domain realm should already exist (from initial -->
    <!-- setup of domain), hence action=" -->
    <realm id='domain' type='tsf:domain' action=">

      <!-- interface resources -->
      <resource id='192.168.100.163' type='interface'>
        <property name='name'      value='if1'/>
        <property name='address'   value='192.168.100.163'/>
        <property name='subnet'    value='255.255.0.0'/>
        <property name='state'     value='1'/>
        <property name='nodekey'   value='192.168.100.163:10000'/>
      </resource>
      <resource id='200.1.1.1' type='interface'>
        <property name='name'      value='if2'/>
        <property name='address'   value='200.1.1.1'/>
        <property name='subnet'    value='255.255.0.0'/>
        <property name='state'     value='1'/>
        <property name='nodekey'   value='192.168.100.163:10000'/>
      </resource>
      <resource id='192.168.100.164' type='interface'>
        <property name='name'      value='if1'/>
        <property name='address'   value='192.168.100.164'/>
        <property name='subnet'    value='255.255.0.0'/>
        <property name='state'     value='1'/>
        <property name='nodekey'   value='192.168.100.164:10000'/>
      </resource>
      <resource id='200.1.1.2' type='interface'>
        <property name='name'      value='if2'/>
        <property name='address'   value='200.1.1.2'/>
        <property name='subnet'    value='255.255.0.0'/>
        <property name='state'     value='1'/>
        <property name='nodekey'   value='192.168.100.164:10000'/>
      </resource>
      <resource id='192.168.100.162' type='interface'>
        <property name='name'      value='if1'/>
        <property name='address'   value='192.168.100.162'/>
        <property name='subnet'    value='255.255.0.0'/>
        <property name='state'     value='1'/>
        <property name='nodekey'   value='192.168.100.162:10000'/>
      </resource>
      <resource id='200.1.1.3' type='interface'>
        <property name='name'      value='if2'/>
        <property name='address'   value='200.1.1.3'/>
        <property name='subnet'    value='255.255.0.0'/>
        <property name='state'     value='1'/>
        <property name='nodekey'   value='192.168.100.162:10000'/>
      </resource>

      <!-- health resources -->
      <resource id='192.168.100.163:10000' type='health'>
        <property name='health'    value='0'/>
      </resource>
      <resource id='192.168.100.164:10000' type='health'>
        <property name='health'    value='0'/>
      </resource>
      <resource id='192.168.100.162:10000' type='health'>
        <property name='health'    value='0'/>
      </resource>

      <!-- cluster resources -->
      <resource id='cluster-1' type='cluster'>
        <property name='type'      value='hot'/>
        <property name='name'      value='cluster-1'/>
        <property name='active'    value='192.168.100.163:10000'/>
        <property name='standby'   value='192.168.100.164:10000'/>
      </resource>

      <!-- configuration resources -->
      <resource id='192.168.100.163:10000' type='configuration'>
        <property name='configuration'   value='{name rc-1}'/>
      </resource>
      <resource id='192.168.100.164:10000' type='configuration'>
        <property name='configuration'   value='{name rc-2}'/>
      </resource>
      <resource id='192.168.100.162:10000' type='configuration'>
        <property name='configuration'   value='{name dir-1}'/>
      </resource>

      <!-- nodes -->
      <node id='192.168.100.163:10000' active='true'>
        <resourceid id='aracf'              type='role'/>
        <resourceid id='aracf'              type='configuration'/>
        <resourceid id='cluster-1'           type='cluster'/>
        <resourceid id='200.1.1.1'           type='interface'/>
```

```
      <resourceid id='192.168.100.163'          type='interface'/>
      <resourceid id='192.168.100.163:10000'    type='health'/>
      <resourceid id='192.168.100.163:10000'    type='configuration'/>
    </node>
    <node id='192.168.100.164:10000' active='true'>
      <resourceid id='aracf'                    type='role'/>
      <resourceid id='aracf'                    type='configuration'/>
      <resourceid id='cluster-1'                type='cluster'/>
      <resourceid id='200.1.1.2'                type='interface'/>
      <resourceid id='192.168.100.164'          type='interface'/>
      <resourceid id='192.168.100.164:10000'    type='health'/>
      <resourceid id='192.168.100.164:10000'    type='configuration'/>
    </node>
    <node id='192.168.100.162:10000' active='true'>
      <resourceid id='spdf'                     type='role'/>
      <resourceid id='spdf'                     type='configuration'/>
      <resourceid id='200.1.1.3'                type='interface'/>
      <resourceid id='192.168.100.162'          type='interface'/>
      <resourceid id='192.168.100.162:10000'    type='health'/>
      <resourceid id='192.168.100.162:10000'    type='configuration'/>
    </node>

    <!-- links -->
    <link id='cluster-1' src='192.168.100.163:10000'
     dst='192.168.100.164:10000' active='true'/>

  </realm>
</tim>



<!-- Director (tds_taf_dir_ services auto-distribute to all directors) -->
<tim realms='tds_taf_dir_realms' resources='tds_taf_dir_resources'
 nodes='tds_taf_dir_nodes' links='tds_taf_dir_links'>
  <!-- example domain information: 2 realms r1_1_1_1 r2_2_2_2 -->

  <!-- app1 realm should already exist (from initial setup of -->
  <!-- domain), hence action='' -->
  <realm id='app1' type='tsf:director' action=''>

    <!-- aracf resources -->
    <resource id='r1_1_1_1' type='aracf'>
      <property name='health'    value='0'/>
      <property name='host'      value='192.168.100.163'/>
      <property name='port'      value='10100'/>
      <property name='qos'       value='aracf_sif'/>
      <property name='saf'       value='aracf_saf'/>
      <property name='fof'       value='aracf_resiliency'/>
      <property name='_health'   value='0'/>
      <property name='_host'     value='192.168.100.164'/>
      <property name='_port'     value='101000'/>
      <property name='_qos'      value='aracf_sif'/>
      <property name='_saf'      value='aracf_saf'/>
      <property name='_fof'      value='aracf_resiliency'/>
    </resource>
    <resource id='r2_2_2_2' type='aracf'>
      <property name='health'    value='0'/>
      <property name='host'      value='192.168.100.163'/>
      <property name='port'      value='10100'/>
      <property name='qos'       value='aracf_sif'/>
      <property name='saf'       value='aracf_saf'/>
      <property name='fof'       value='aracf_resiliency'/>
      <property name='_health'   value='0'/>
      <property name='_host'     value='192.168.100.164'/>
      <property name='_port'     value='101000'/>
      <property name='_qos'      value='aracf_sif'/>
      <property name='_saf'      value='aracf_saf'/>
      <property name='_fof'      value='aracf_resiliency'/>
    </resource>

    <!-- pool resources -->
    <resource id='1.1.1.1/24/app1' type='pool'>
      <property name='ip'     value='1.1.1.1'/>
      <property name='mask'   value='24'/>
      <property name='vpn'    value='app1'/>
      <property name='data'   value='r1_1_1_1'/>
    </resource>
    <resource id='2.2.2.2/24/app1' type='pool'>
      <property name='ip'     value='2.2.2.2'/>
      <property name='mask'   value='24'/>
      <property name='vpn'    value='app1'/>
      <property name='data'   value='r2_2_2_2'/>
    </resource>

    <!-- nodes -->
    <node id='r1_1_1_1' active='true'>
      <resourceid id='r1_1_1_1'          type='aracf'/>
      <resourceid id='1.1.1.1/24/app1' type='pool'/>
    </node>
    <node id='r2_2_2_2' active='true'>
      <resourceid id='r2_2_2_2'          type='aracf'/>
      <resourceid id='2.2.2.2/24/app1' type='pool'/>
    </node>

  </realm>
```

```
</tim>


<!-- Resource Realms (for RC's), (tds_taf_rc services push all -->
<!-- updates to any assigned RC's [best effort]) -->
<tim realms='tds_manage_realms' resources='tds_taf_rc_resources'
 nodes='tds_taf_rc_nodes' links='tds_taf_rc_links'>

  <!-- realm r1_1_1_1, has one BRAS and 8 HGW's-->
  <!--  r1_1_1_1 -->
  <!--    port 0 -->
  <!--       vpi 0 -->
  <!--          vci 0 -->
  <!--          vci 1 -->
  <!--       vpi 1 -->
  <!--          vci 0 -->
  <!--          vci 1 -->
  <!--    port 1 -->
  <!--       vpi 0 -->
  <!--          vci 0 -->
  <!--          vci 1 -->
  <!--       vpi 1 -->
  <!--          vci 0 -->
  <!--          vci 1 -->
  <!--  vp, vc resources for upstream (-us) and downstream (-ds) links -->

  <realm id='r1_1_1_1' type='tsf:realm' action=''>
    <!-- VP's -->
    <resource id='r1_1_1_1-0-0-us' type='vp'>
      <property name='used' value='0'/>
      <property name='capacity' value='64000'/>
    </resource>
    <resource id='r1_1_1_1-0-0-ds' type='vp'>
      <property name='used' value='0'/>
      <property name='capacity' value='64000'/>
    </resource>
    <resource id='r1_1_1_1-0-1-us' type='vp'>
      <property name='used' value='0'/>
      <property name='capacity' value='64000'/>
    </resource>
    <resource id='r1_1_1_1-0-1-ds' type='vp'>
      <property name='used' value='0'/>
      <property name='capacity' value='64000'/>
    </resource>
    <resource id='r1_1_1_1-1-0-us' type='vp'>
      <property name='used' value='0'/>
      <property name='capacity' value='64000'/>
```

```
</resource>
<resource id='r1_1_1_1-1-0-ds' type='vp'>
  <property name='used' value='0'/>
  <property name='capacity' value='64000'/>
</resource>
<resource id='r1_1_1_1-1-1-us' type='vp'>
  <property name='used' value='0'/>
  <property name='capacity' value='64000'/>
</resource>
<resource id='r1_1_1_1-1-1-ds' type='vp'>
  <property name='used' value='0'/>
  <property name='capacity' value='64000'/>
</resource>

<!-- VC's -->
<resource id='r1_1_1_1-0-0-0-us' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-0-0-0-ds' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-0-0-1-us' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-0-0-1-ds' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-0-1-0-ds' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-0-1-0-us' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-1-0-0-us' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-1-0-0-ds' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
```

```
<resource id='r1_1_1_1-0-1-1-us' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-0-1-1-ds' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-1-0-1-us' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-1-0-1-ds' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-1-1-0-us' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-1-1-0-ds' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-1-1-1-us' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r1_1_1_1-1-1-1-ds' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>

<!-- nodes -->
<node id='r1_1_1_1' active='true'/>
<node id='r1_1_1_1-0-0-0' active='true'/>
<node id='r1_1_1_1-0-0-1' active='true'/>
<node id='r1_1_1_1-0-1-0' active='true'/>
<node id='r1_1_1_1-0-1-1' active='true'/>
<node id='r1_1_1_1-1-0-0' active='true'/>
<node id='r1_1_1_1-1-0-1' active='true'/>
<node id='r1_1_1_1-1-1-0' active='true'/>
<node id='r1_1_1_1-1-1-1' active='true'/>

<!-- links -->
<link id='r1_1_1_1=r1_1_1_1-0-0-0' src='r1_1_1_1'
 dst='r1_1_1_1-0-0-0' active='true'>
```

```
  <resourceid id='r1_1_1_1-0-0-ds' type='vp'/>
  <resourceid id='r1_1_1_1-0-0-0-ds' type='vc'/>
</link>
<link id='r1_1_1_1-0-0-0=r1_1_1_1' src='r1_1_1_1-0-0-0'
 dst='r1_1_1_1' active='true'>
  <resourceid id='r1_1_1_1-0-0-us' type='vp'/>
  <resourceid id='r1_1_1_1-0-0-0-us' type='vc'/>
</link>
<link id='r1_1_1_1=r1_1_1_1-0-0-1' src='r1_1_1_1'
 dst='r1_1_1_1-0-0-1' active='true'>
  <resourceid id='r1_1_1_1-0-0-ds' type='vp'/>
  <resourceid id='r1_1_1_1-0-0-1-ds' type='vc'/>
</link>
<link id='r1_1_1_1-0-0-1=r1_1_1_1' src='r1_1_1_1-0-0-1'
 dst='r1_1_1_1' active='true'>
  <resourceid id='r1_1_1_1-0-0-us' type='vp'/>
  <resourceid id='r1_1_1_1-0-0-1-us' type='vc'/>
</link>
<link id='r1_1_1_1=r1_1_1_1-0-1-0' src='r1_1_1_1'
 dst='r1_1_1_1-0-1-0' active='true'>
  <resourceid id='r1_1_1_1-0-1-ds' type='vp'/>
  <resourceid id='r1_1_1_1-0-1-0-ds' type='vc'/>
</link>
<link id='r1_1_1_1-0-1-0=r1_1_1_1' src='r1_1_1_1-0-1-0'
 dst='r1_1_1_1' active='true'>
  <resourceid id='r1_1_1_1-0-1-us' type='vp'/>
  <resourceid id='r1_1_1_1-0-1-0-us' type='vc'/>
</link>
<link id='r1_1_1_1=r1_1_1_1-0-1-1' src='r1_1_1_1'
 dst='r1_1_1_1-0-1-1' active='true'>
  <resourceid id='r1_1_1_1-0-1-ds' type='vp'/>
  <resourceid id='r1_1_1_1-0-1-1-ds' type='vc'/>
</link>
<link id='r1_1_1_1-0-1-1=r1_1_1_1' src='r1_1_1_1-0-1-1'
 dst='r1_1_1_1' active='true'>
  <resourceid id='r1_1_1_1-0-1-us' type='vp'/>
  <resourceid id='r1_1_1_1-0-1-1-us' type='vc'/>
</link>
<link id='r1_1_1_1=r1_1_1_1-1-0-0' src='r1_1_1_1'
 dst='r1_1_1_1-1-0-0' active='true'>
  <resourceid id='r1_1_1_1-1-0-ds' type='vp'/>
  <resourceid id='r1_1_1_1-1-0-0-ds' type='vc'/>
</link>
<link id='r1_1_1_1-1-0-0=r1_1_1_1' src='r1_1_1_1-1-0-0'
 dst='r1_1_1_1' active='true'>
  <resourceid id='r1_1_1_1-1-0-us' type='vp'/>
  <resourceid id='r1_1_1_1-1-0-0-us' type='vc'/>
  </link>
  <link id='r1_1_1_1=r1_1_1_1-1-0-1' src='r1_1_1_1'
   dst='r1_1_1_1-1-0-1' active='true'>
     <resourceid id='r1_1_1_1-1-0-ds' type='vp'/>
     <resourceid id='r1_1_1_1-1-0-1-ds' type='vc'/>
  </link>
  <link id='r1_1_1_1-1-0-1=r1_1_1_1' src='r1_1_1_1-1-0-1'
   dst='r1_1_1_1' active='true'>
     <resourceid id='r1_1_1_1-1-0-us' type='vp'/>
     <resourceid id='r1_1_1_1-1-0-1-us' type='vc'/>
  </link>
  <link id='r1_1_1_1=r1_1_1_1-1-1-0' src='r1_1_1_1'
   dst='r1_1_1_1-1-1-0' active='true'>
     <resourceid id='r1_1_1_1-1-1-ds' type='vp'/>
     <resourceid id='r1_1_1_1-1-1-0-ds' type='vc'/>
  </link>
  <link id='r1_1_1_1-1-1-0=r1_1_1_1' src='r1_1_1_1-1-1-0'
   dst='r1_1_1_1' active='true'>
     <resourceid id='r1_1_1_1-1-1-us' type='vp'/>
     <resourceid id='r1_1_1_1-1-1-0-us' type='vc'/>
  </link>
  <link id='r1_1_1_1=r1_1_1_1-1-1-1' src='r1_1_1_1'
   dst='r1_1_1_1-1-1-1' active='true'>
     <resourceid id='r1_1_1_1-1-1-ds' type='vp'/>
     <resourceid id='r1_1_1_1-1-1-1-ds' type='vc'/>
  </link>
  <link id='r1_1_1_1-1-1-1=r1_1_1_1' src='r1_1_1_1-1-1-1'
   dst='r1_1_1_1' active='true'>
     <resourceid id='r1_1_1_1-1-1-us' type='vp'/>
     <resourceid id='r1_1_1_1-1-1-1-us' type='vc'/>
  </link>
</realm>



<!-- realm r2_2_2_2, has one BRAS and 8 HGW's-->
<!--  r1_1_1_1 -->
<!--    port 0 -->
<!--       vpi 0 -->
<!--          vci 0 -->
<!--          vci 1 -->
<!--       vpi 1 -->
<!--          vci 0 -->
<!--          vci 1 -->
<!--    port 1 -->
<!--       vpi 0 -->
<!--          vci 0 -->
<!--          vci 1 -->
```

```
<!--       vpi 1 -->
<!--          vci 0 -->
<!--          vci 1 -->
<!--  vp, vc resources for upstream (-us) and downstream (-ds) links -->

<realm id='r2_2_2_2' type='tsf:realm' action=''>
  <!-- VP's -->
  <resource id='r2_2_2_2-0-0-us' type='vp'>
    <property name='used' value='0'/>
    <property name='capacity' value='64000'/>
  </resource>
  <resource id='r2_2_2_2-0-0-ds' type='vp'>
    <property name='used' value='0'/>
    <property name='capacity' value='64000'/>
  </resource>
  <resource id='r2_2_2_2-0-1-us' type='vp'>
    <property name='used' value='0'/>
    <property name='capacity' value='64000'/>
  </resource>
  <resource id='r2_2_2_2-0-1-ds' type='vp'>
    <property name='used' value='0'/>
    <property name='capacity' value='64000'/>
  </resource>
  <resource id='r2_2_2_2-1-0-us' type='vp'>
    <property name='used' value='0'/>
    <property name='capacity' value='64000'/>
  </resource>
  <resource id='r2_2_2_2-1-0-ds' type='vp'>
    <property name='used' value='0'/>
    <property name='capacity' value='64000'/>
  </resource>
  <resource id='r2_2_2_2-1-1-us' type='vp'>
    <property name='used' value='0'/>
    <property name='capacity' value='64000'/>
  </resource>
  <resource id='r2_2_2_2-1-1-ds' type='vp'>
    <property name='used' value='0'/>
    <property name='capacity' value='64000'/>
  </resource>

  <!-- VC's -->
  <resource id='r2_2_2_2-0-0-0-us' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-0-0-0-ds' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-0-0-1-us' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-0-0-1-ds' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-0-1-0-us' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-0-1-0-ds' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-0-1-1-us' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-0-1-1-ds' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-1-0-0-us' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-1-0-0-ds' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-1-0-1-us' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-1-0-1-ds' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-1-1-0-us' type='vc'>
    <property name='used' value='0'/>
    <property name='capacity' value='32000'/>
  </resource>
  <resource id='r2_2_2_2-1-1-0-ds' type='vc'>
```

```
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r2_2_2_2-1-1-1-us' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>
<resource id='r2_2_2_2-1-1-1-ds' type='vc'>
  <property name='used' value='0'/>
  <property name='capacity' value='32000'/>
</resource>

<!-- nodes -->
<node id='r2_2_2_2' active='true'/>
<node id='r2_2_2_2-0-0-0' active='true'/>
<node id='r2_2_2_2-0-0-1' active='true'/>
<node id='r2_2_2_2-0-1-0' active='true'/>
<node id='r2_2_2_2-0-1-1' active='true'/>
<node id='r2_2_2_2-1-0-0' active='true'/>
<node id='r2_2_2_2-1-0-1' active='true'/>
<node id='r2_2_2_2-1-1-0' active='true'/>
<node id='r2_2_2_2-1-1-1' active='true'/>

<!-- links -->
<link id='r2_2_2_2=r2_2_2_2-0-0-0' src='r2_2_2_2'
 dst='r2_2_2_2-0-0-0' active='true'>
  <resourceid id='r2_2_2_2-0-0-ds' type='vp'/>
  <resourceid id='r2_2_2_2-0-0-0-ds' type='vc'/>
</link>
<link id='r2_2_2_2-0-0-0=r2_2_2_2' src='r2_2_2_2-0-0-0'
 dst='r2_2_2_2' active='true'>
  <resourceid id='r2_2_2_2-0-0-us' type='vp'/>
  <resourceid id='r2_2_2_2-0-0-0-us' type='vc'/>
</link>
<link id='r2_2_2_2=r2_2_2_2-0-0-1' src='r2_2_2_2'
 dst='r2_2_2_2-0-0-1' active='true'>
  <resourceid id='r2_2_2_2-0-0-ds' type='vp'/>
  <resourceid id='r2_2_2_2-0-0-1-ds' type='vc'/>
</link>
<link id='r2_2_2_2-0-0-1=r2_2_2_2' src='r2_2_2_2-0-0-1'
 dst='r2_2_2_2' active='true'>
  <resourceid id='r2_2_2_2-0-0-us' type='vp'/>
  <resourceid id='r2_2_2_2-0-0-1-us' type='vc'/>
</link>
<link id='r2_2_2_2=r2_2_2_2-0-1-0' src='r2_2_2_2'
 dst='r2_2_2_2-0-1-0' active='true'>
  <resourceid id='r2_2_2_2-0-1-ds' type='vp'/>
```

```
   <resourceid id='r2_2_2_2-0-1-0-ds' type='vc'/>
</link>
<link id='r2_2_2_2-0-1-0=r2_2_2_2' src='r2_2_2_2-0-1-0'
 dst='r2_2_2_2' active='true'>
   <resourceid id='r2_2_2_2-0-1-us' type='vp'/>
   <resourceid id='r2_2_2_2-0-1-0-us' type='vc'/>
</link>
<link id='r2_2_2_2=r2_2_2_2-0-1-1' src='r2_2_2_2'
 dst='r2_2_2_2-0-1-1' active='true'>
   <resourceid id='r2_2_2_2-0-1-ds' type='vp'/>
   <resourceid id='r2_2_2_2-0-1-1-ds' type='vc'/>
</link>
<link id='r2_2_2_2-0-1-1=r2_2_2_2' src='r2_2_2_2-0-1-1'
 dst='r2_2_2_2' active='true'>
   <resourceid id='r2_2_2_2-0-1-us' type='vp'/>
   <resourceid id='r2_2_2_2-0-1-1-us' type='vc'/>
</link>
<link id='r2_2_2_2=r2_2_2_2-1-0-0' src='r2_2_2_2'
 dst='r2_2_2_2-1-0-0' active='true'>
   <resourceid id='r2_2_2_2-1-0-ds' type='vp'/>
   <resourceid id='r2_2_2_2-1-0-0-ds' type='vc'/>
</link>
<link id='r2_2_2_2-1-0-0=r2_2_2_2' src='r2_2_2_2-1-0-0'
 dst='r2_2_2_2' active='true'>
   <resourceid id='r2_2_2_2-1-0-us' type='vp'/>
   <resourceid id='r2_2_2_2-1-0-0-us' type='vc'/>
</link>
<link id='r2_2_2_2=r2_2_2_2-1-0-1' src='r2_2_2_2'
 dst='r2_2_2_2-1-0-1' active='true'>
   <resourceid id='r2_2_2_2-1-0-ds' type='vp'/>
   <resourceid id='r2_2_2_2-1-0-1-ds' type='vc'/>
</link>
<link id='r2_2_2_2-1-0-1=r2_2_2_2' src='r2_2_2_2-1-0-1'
 dst='r2_2_2_2' active='true'>
   <resourceid id='r2_2_2_2-1-0-us' type='vp'/>
   <resourceid id='r2_2_2_2-1-0-1-us' type='vc'/>
</link>
<link id='r2_2_2_2=r2_2_2_2-1-1-0' src='r2_2_2_2'
 dst='r2_2_2_2-1-1-0' active='true'>
   <resourceid id='r2_2_2_2-1-1-ds' type='vp'/>
   <resourceid id='r2_2_2_2-1-1-0-ds' type='vc'/>
</link>
<link id='r2_2_2_2-1-1-0=r2_2_2_2' src='r2_2_2_2-1-1-0'
 dst='r2_2_2_2' active='true'>
   <resourceid id='r2_2_2_2-1-1-us' type='vp'/>
   <resourceid id='r2_2_2_2-1-1-0-us' type='vc'/>
</link>
```

```
      <link id='r2_2_2_2=r2_2_2_2-1-1-1' src='r2_2_2_2'
       dst='r2_2_2_2-1-1-1' active='true'>
        <resourceid id='r2_2_2_2-1-1-ds' type='vp'/>
        <resourceid id='r2_2_2_2-1-1-1-ds' type='vc'/>
      </link>
      <link id='r2_2_2_2-1-1-1=r2_2_2_2' src='r2_2_2_2-1-1-1'
       dst='r2_2_2_2' active='true'>
        <resourceid id='r2_2_2_2-1-1-us' type='vp'/>
        <resourceid id='r2_2_2_2-1-1-1-us' type='vc'/>
      </link>
    </realm>

  </tim>

</global>
```

5

Service Control Engine

Overview

The Service Control Engine (SCE) provides an integration with the Cisco Deep Packet Inspection (DPI) platform (Cisco SCE). As part of the Cisco Service Execution Framework (SEF) initiative, DPI technology delivers and deploys per subscriber application management services integrated with the Policy Management solution. SCE 3.0 integration with the BPM is a key component of the SEF architecture.

The SCE Subscriber application program interface (API) module enables subscriber management integrations with the SCE. The API is generic and independent of the protocol in use between the policy server and the SCE. This allows the user to implement the API over another protocol (for example, SOAP/BEEP). The solution includes the SCE Subscriber API implementation over Proprietary Remote Procedure Call (PRPC) protocol.

Two modes of operations work with the SCE 3.0:

- Package Push
- Event Login Pull Notification

Package Push

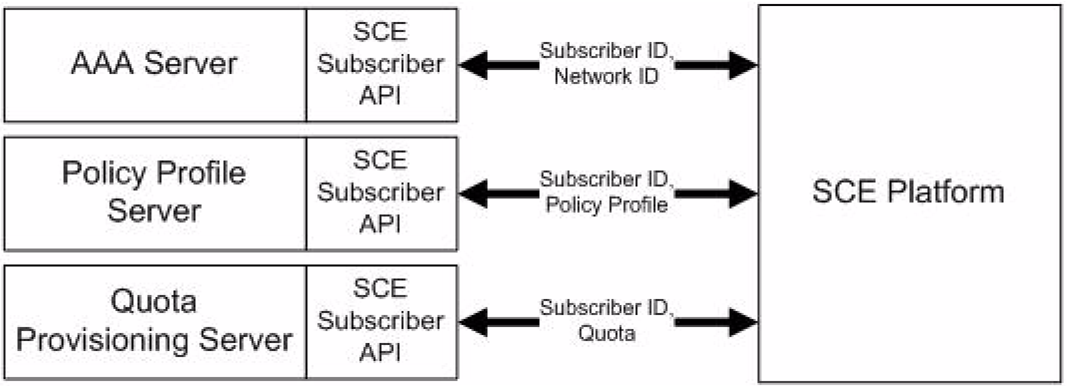
In the Package Push mode, the BPM orchestrates the assignment of per subscriber package, based on an (non-SCE initiated) external trigger. The trigger could be a subscriber accessing a portal, subscribing to a service, or pushing a trigger to the BPM, which pushes a package to the SCE for the subscriber session.

Event Login Pull Notification

In the Event Login Pull notification mode, the SCE initiates the request for a policy to be applied on it. Here, an event initiated at the SCE invokes the SCE to request the package information from the BPM. This event could be the initiation of a new IP session appearing at the SCE. Here, the *Pull* action is a *Login* event that articulates to the BPM that a new session is established and requests a corresponding profile (package) from the BPM.

The SCE Subscriber API allows external applications (policy servers) to connect to the SCE directly for subscriber provisioning. The subscriber provisioning process updates the network IDs, policy profile, and quota characteristics of the subscriber using the subscriber ID. The API can be installed, and used concurrently, on several policy servers. Each server can perform different parts of the subscriber provisioning process, as shown in Figure 15.

Figure 15. API Use by Several Servers



The API uses the PRPC protocol as a transport for the connection to the SCE. The API implementation supplied has the following limitations:

- Java implementation only
- Every API instance supports a connection to exactly one SCE platform

Features

The BPM establishes the SCE connection. From a BPM system perspective, for any devices for which the system tracks sessions, the existing capacity admission control functionality works with it.

- Implements SCE 3.0 API
- Supports SCE IP Session awareness
- Supports Session Removal
- Failover considerations (Primary / Secondary Resource Controller failover)
- Supports SCE/BPM Synchronization
- Supports SCE 3.0 device configuration capability within the BPM system
- Supports package pushes (unsolicited by the SCE)

SCE API Functions

The BPM supports the following SCE API functions:

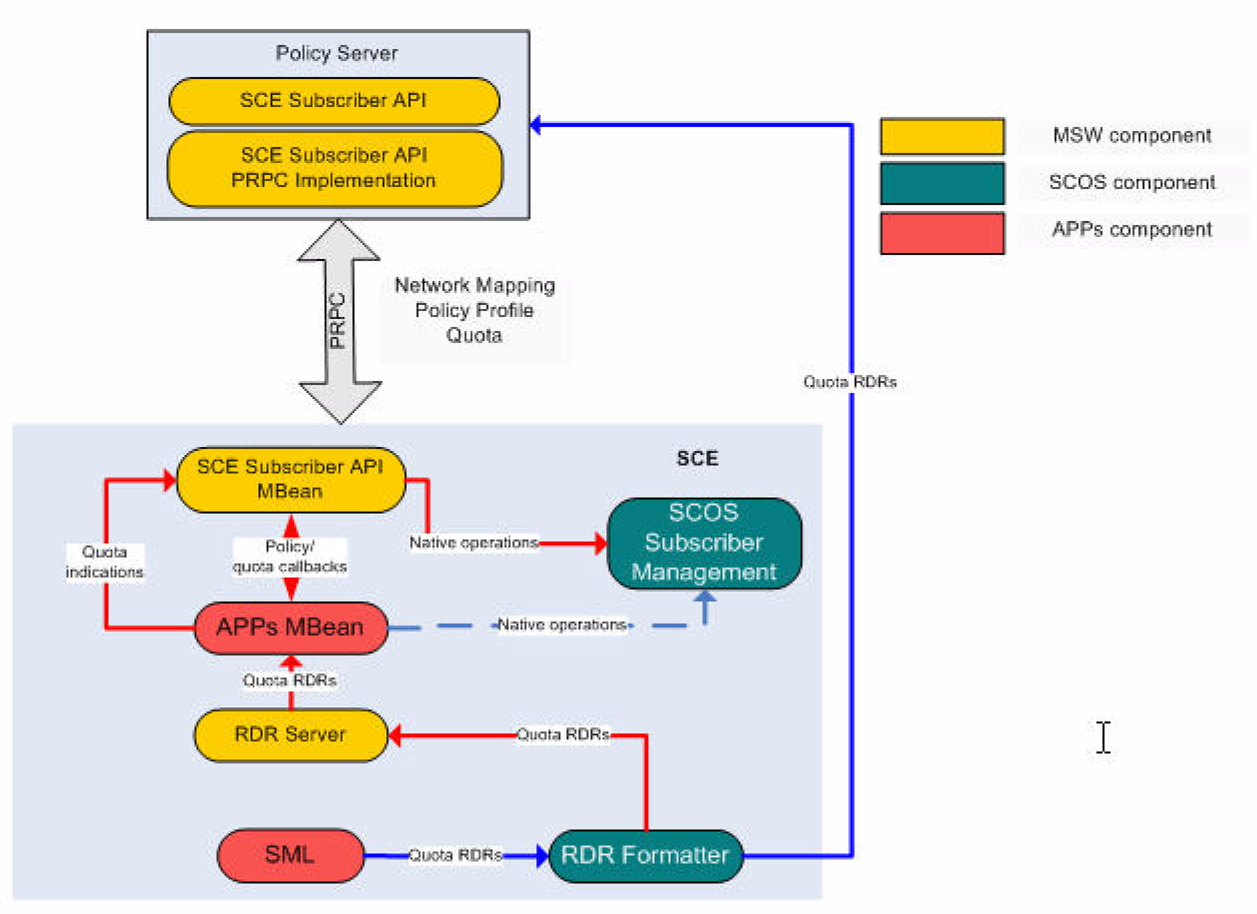
- Login-Pull Event Handling
- Logout Event Handling
- Policy Package Push
- Keep Alive Mechanism

Architecture

The SCE Subscriber API solution consists of the following components, depicted in Figure 16:

- Policy Server Components
- Platform Components

Figure 16. SCE Subscriber API Architecture



Policy Server Components

The generic SCE Subscriber API framework provides integration directly with the SCE (without SM). The framework does not rely on a specific transport protocol. It provides an interface for the implementation over different protocols.

The SCE Subscriber API PRPC implementation interacts with the components residing on the SCE platform via the PRPC protocol. It activates subscriber operations on the SCE platform and receives indications from the SCE platform.

Platform Components

The following are the implemented platform components:

- *SCESubscriberAPI MBean* - This component resides on the SCE platform as part of the SCE agent. It performs and dispatches SCE Subscriber API requests and commands. It also activates native subscriber operations in the SCOS.

- *APPs MBean* **-** This component represents an application *MBean* that resides at the SCE platform as part of the SCE Agent. It is responsible for policy and quota customization operations. It interacts with *SCESubscriberAPI MBean* via callback methods. It receives subscriber information, processes it, and returns it to the *SCESubscriberAPI MBean* for further processing. In addition, it can activate native subscriber operations in the SCOS. This component can also receive Quota RDRs from the RDR server residing on the SCE platform, convert them to Quota indications, and pass them to the *SCESubscriberAPI MBean* for further processing.
- SCOS Subscriber Management **-** This component represents the SCOS software used for subscriber management calls
- RDR Formatter **-** This component represents the SCOS software module that the SML application uses for RDR forwarding.
- RDR Server **-** This component, existing at the SCE platform as part of the SCE Agent, receives RDRs from the SML.
- SML **-** This application component sends RDRs directly, via RDR Formatter, to *outside world* components that can receive the RDRs.

It is possible to implement SCE Subscriber API interface over a different protocol. This requires implementing the support of this protocol at the SCE (similar to PRPC protocol support) without changing the API logic at the SCE.

Supported Topologies

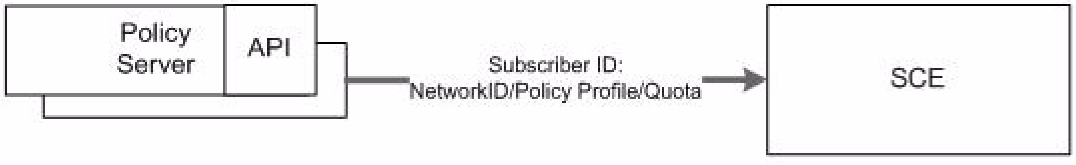
The following are the recommended topologies:

- One Policy Server
- Two Policy Servers Topology
- Three Policy Servers Topology
- DHCP Lease Query LEG Topology
- SCMS-SM Topology

One Policy Server

In the one policy server or two-node cluster topology (Figure 17), one server handles all aspects of the subscriber provisioning process.

Figure 17. One Policy Server



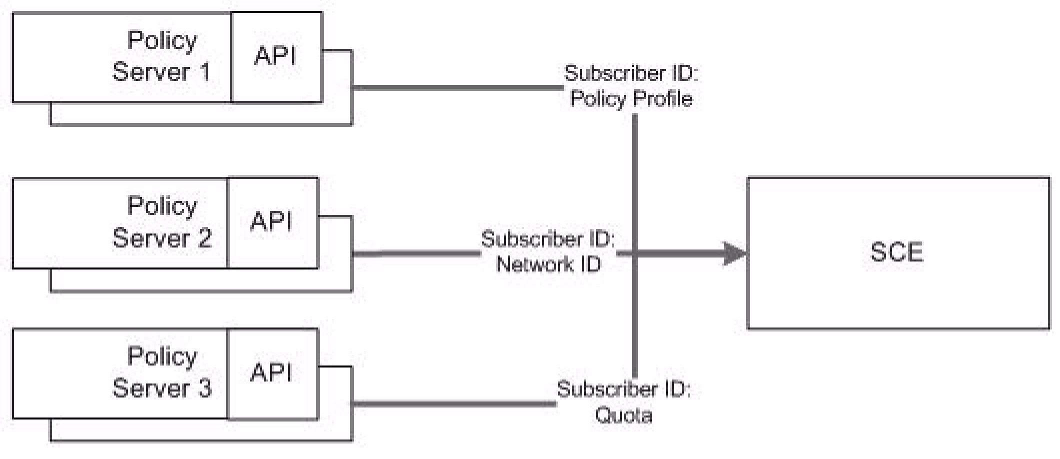
Two Policy Servers Topology

In the two policy servers or two two-node clusters topology Figure 17), one server is responsible for two aspects of the subscriber provisioning. The other server is responsible for only one aspect. Here, any combination is allowed.

Three Policy Servers Topology

In the three policy servers or three two-node clusters topology (Figure 18), each server is responsible for a different aspect of the subscriber provisioning process:

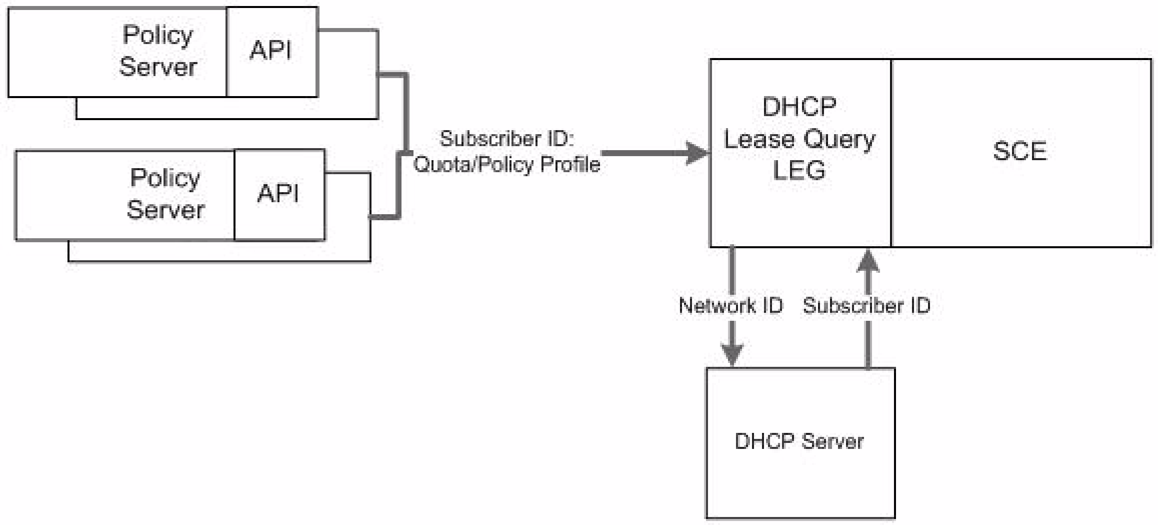
Figure 18. Three Policy Servers Topology



DHCP Lease Query LEG Topology

The DHCP Lease Query LEG topology (Figure 19) maps a Network ID to a Subscriber ID with one or more policy servers, as depicted in the three policy server example in Figure 18.

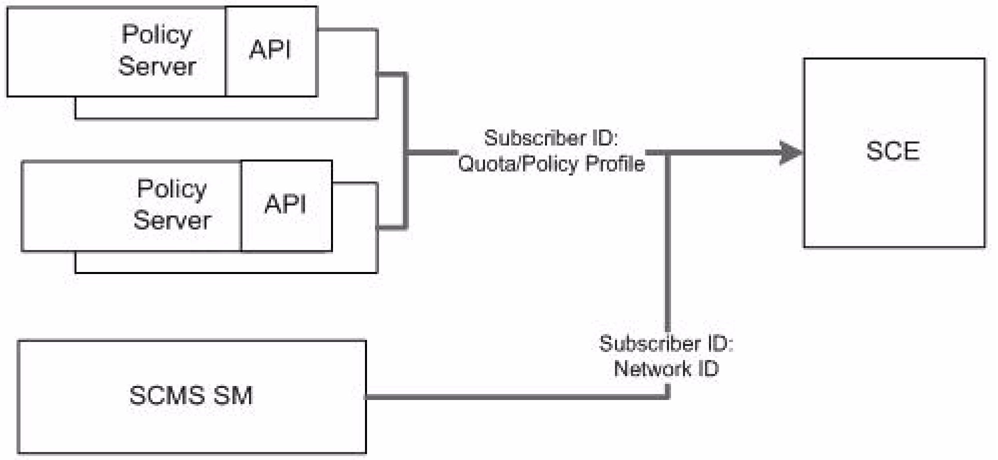
Figure 19. DHCP Lease Query LEG Topology



SCMS-SM Topology

The SCMS-SM topology (Figure 20) maps Network ID to Subscriber ID with one or more policy servers. The number of policy servers depends on whether the SM is used for policy profile provisioning, in addition to the network ID.

Figure 20. SCMS-SM Topology


Policy Server
API
Policy Server
API
Subscriber ID: Quota/Policy Profile
SCE
Subscriber ID: Network ID
SCMS SM

Note: Use caution when using more than one policy server for the same provisioning purpose. If the SCE platform receives different subscriber information from two policy servers, a loss of synchronization occurs with at least one of the policy servers. If two policy servers provide Subscriber ID/ Network ID correlation for the same subscriber, the SCE is always synchronized with the policy server that performed the *last* subscriber update. The API does not limit any topology use. However, the SCE platform does not correlate between all policy servers performing subscriber provisioning.

High Availability

High availability API support assumes that the high-availability scheme of the policy server is a two-node cluster in which only one server is active at a time and the standby server is not connected to the SCE. When the active server fails, the two-node cluster performs a failover to the standby server.

Note: High-availability can be implemented separately for each policy server provisioning the SCE platform at the same time.

Procedure: Implementing High Availability

To implement high-availability with the SCE Subscriber API perform the following procedure:

1. Set up a two-node cluster for two policy servers.
2. Construct two API instances with the same API.
3. Name each one on the different server (node) within the cluster.

During cluster runtime, only one API instance can be connected to the SCE platform. When failover occurs, the failed server disconnects from the SCE. The standby server becomes active and reconnects to the SCE within a predefined timeout. If identical API names exist, the SCE behaves as if the same API was reconnected. No information is lost.

Use Cases

This section discusses the following use cases:

- Session Establishment
- Session Termination
- Policy Push
- SCE Cluster Member Failure

Session Establishment

This section discusses a use case in which the BPM responds to a new session appearing on the network. In this case, the network session establishment is the Primary SAF source and the SCE is the secondary SAF source. The network presence events occur prior to the SCE login session event.

Session Establishment Use Case Sequence

1. The subscriber initiates a network session
2. A presence event is forwarded to the presence function (DHCP, RADIUS, etc.) of the service provider.
3. The presence function forwards the policy server.
4. The BPM adds the user session and determines the default SCE policy.
5. The session is captured at the SCE (anonymous group).
6. The SCE generates a Login-Pull-Request to the BPM.
7. The BPM adds the SCE id into the session table (SSF).
8. The BPM initiates Login-Pull-Response to SCE that includes the required package (*profile-B*).

Note: A race condition exists in the sequence involving step 3-4 and step 5. If the SCE generates a Login Pull Request prior to the Presence Function notifying the BPM, the BPM does not respond to the SCE and wait for the SCE to retransmit a pull request.

Session Termination

This section discusses a use case in which the BPM responds to a session disconnection on the network. This is driven by the primary SAF source. In this case, it is driven by the network presence SAF.

Session Termination Use Case Sequence

1. The user disconnects from the network (PPP session cleared down/Radius Stop, DHCP Lease expiry).

2. The network elements signal to presence function.
3. The presence function signals to BPM.
4. The BPM removes the user session.
5. The BPM initiates a Logout-Push-Request to the SCE.
6. The SCE acknowledges logout request with Logout-Pull-Response.

Policy Push

This section discusses a use case in which the BPM handles a portal-driven package change request.

Policy Push Use Case Sequence

1. The user Logs into Self Management Portal.
2. The portal requests current policy state from BPM.
3. The BPM responds with the current state information. The portal displays the real-time settings through the user portal.
4. The user selects a new service and instructs the BPM of the selected service.
5. The BPM determines policy behavior.
6. The BPM instructs the relevant SCE to install a new package against the subscriber.
7. The SCE responds to the package request to confirm that it has received the request and is applying the new package.
8. The subscriber current package assignment state is updated.
9. The BPM informs the Portal that the request is successful.

SCE Cluster Member Failure

This section discusses a use case following the same logic as the Session Establishment and Session Termination cases, using further detail. The BPM remembers (with its state transition model) the current subscriber state and package. The new *Login-Request-Response* returns the last know package state.

SCE Cluster Member Failure Sequence

1. The session is established as described in the Session Establishment use case.
2. The SCE 3 fails.
3. The system redirects user traffic to SCE 2 or any other SCE in the cluster.
4. SCE 2 sends a Login-Pull request to the BPM.
5. The BPM finds an existing session record for IP address indicated in the login pull.
6. The BPM obtains the current policy profile assigned in the existing session record.
7. The BPM pushes the policy profile to SCE 2.
8. The BPM pushes a logout event to the SCE.

SCE Log Files

This section discusses default and subscriber operations log messages.

Default Log Messages

The SCE platform provides the ability to log all operations called by the policy server into the SCE user-log file. The SCE issues the following messages by default, without configuration.

- <client-name> - connect operation was called, registered listeners: <type of the listeners that were registered>
- <client-name> - disconnected
- <client-name> - registered a Login Pull Listener
- <client-name> - unregistered a Pull Listener
- <client-name> - registered a Logout Listener
- <client-name> - unregistered a Logout Listener
- <client-name> - registered Quota Listener
- <client-name> - unregister Quota Listener
- <client-name> - synchronize Push Start
- <client-name> - synchronize Push End
- <client-name> - synchronize Pull Start
- <client-name> - synchronize Pull End
- <client-name> - getSceAgentVersion was called

Subscriber Operations Log Messages

The special flag conditions subscriber operations log messages. For troubleshooting purposes, enable this flag to receive those messages. Use the procedure below to manage this message logging.

Procedure: Managing Operations Log Messages

1. To enable logging, enter the following CLI command at the SCE platform:

   config)# **management-agent sce-api logging**

2. To view the log file, enter the following CLI command at the SCE platform:

   #>**logger get user-log <FILE NAME>**

3. To disable the logging use the following CLI at the SCE platform:

   (config)#> **no management-agent sce-api logging**

4. To determine if logging is enabled, use the following CLI command at the SCE platform:

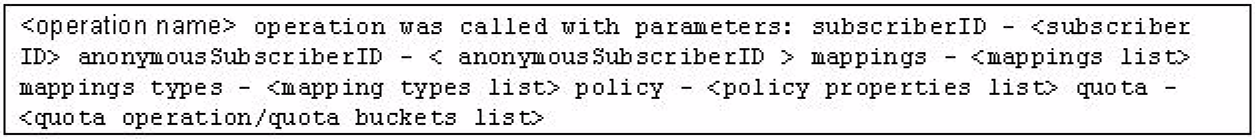
   #> **show management-agent sce-api**

   When the logging flag is enabled, the system issues the message in Figure 21 for the following operations:

   - login operation
   - *networkIDUpdate* operation
   - logout operation
   - *quotaUpdate* operation
   - *loginPullResponse* operation
   - *profileUpdate* operation

- *getQuotaStatus* operation

Figure 21. Logging Flag Message

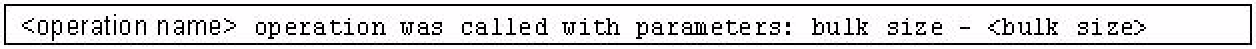


```
<operation name> operation was called with parameters: subscriberID - <subscriber
ID> anonymousSubscriberID - < anonymousSubscriberID > mappings - <mappings list>
mappings types - <mapping types list> policy - <policy properties list> quota -
<quota operation/quota buckets list>
```

The system issues the message in Figure 22 for the following bulk operations:

- *loginBulk* operation
- *networkIDUpdateBulk* operation
- *logoutBulk* operation
- *quotaUpdateBulk* operation
- *loginPullBulkResponse* operation
- *profileUpdateBulk* operation
- *getQuotaStatuBulkRequest* operation
- *getSubscribersBulk*

Figure 22. Bulk Logging Flag



```
<operation name> operation was called with parameters: bulk size - <bulk size>
```

Installation

The SCE Subscriber API is released separately as a ZIP file and includes the following:

- Javadoc
- SCE Subscriber API Programmers Guide
- JAR file containing SCE Subscriber API implementation over PRPC protocol

SCE IP Session Awareness Support

BPM captures two sources of presence information per subscriber session event. A new subscriber session is signaled through a conventional network presence source (for example, RADIUS Accounting). It also captures the subscriber SCE IP mapping, via the *Login-Pull-Request* message.

Dual SAF Support

The BPM supports dual sources for presence (session awareness) per subscriber connection event. The BPM supports the selection of a primary source for the session presence. It also supports a secondary source for the session presence to update the session record in the BPM system. This functionality assists in handling race conditions.

Race Condition

In a typical deployment of SCE within a broadband architecture, the first presence notification from a RADIUS Accounting source is received directly from the BRAS or via RADIUS Proxy. This notifies the BPM of the network session creation. When the network session is created in the network, traffic flows appropriately. This invokes a login event at the SCE. Based on the specific network deployment, the SCE event can appear at the BPM before the Network Presence (RADIUS) event. If this occurs, the BPM can handle this possibility satisfactorily.

Request and Response for Login Pull

When the BPM receives a *Login-Pull-Request*, it responds to the request with a *Login-Pull-Response* message. However, the BPM does not respond to the *Race* condition. In all other circumstances, the BPM responds with a *Login-Pull-Response* message.

The *Login-Pull-Response* contains a valid subscriber ID. The BPM provides the subscriber ID. This is the subscriber username (for example, user@domain), access-line ID (for example, BRAS/Port/VP/VC) or another unique identifier, based on the specific implementation (for example, BBIP from ADQ or Technical Key from TI).

The *Login-Pull-Response* can contain a policy profile classification. This indicates to the SCE the package to deploy against this subscriber session. This may not be necessary if the SCE default is an acceptable global default for IP traffic. In the case of a limited set of default SCE profiles, an service provider may have a limited set of base services. When a subscriber connects, he or she requires the allocation of a package related to that service. This requires a preprovisioned package allocated to the subscriber. This may be defined as a service class within which multiple subscribers may exist. Behind that service class is the real SCE package definition. When it receives the *Login-Pull-Request*, the SSF can trigger a lookup to the subscriber repository to define the default package to apply at the SCE for this subscriber session.

Push the Policy

If a default package is assigned to the subscriber in the policy repository, the system leverages the login response to issue the SCE with the correct package for that subscriber. This includes the ability to set a default SCE package at the BPM.

Subscriber ID

The BPM responds, in the *Login* response, with a subscriber ID. This can relate to the access-line ID (the BRAS port) or to the subscriber username of specific subscriber identity in use within the BPM.

Extend Session Record

The BPM system extends its generic session record to support SCE session mapping.

Session Removal

The system generates a triggered logout event in the following cases:

- Session Removal
- Session Removal for Failover Scenario

Session Removal

The BPM creates a Logout-Request event when the primary SAF source generates a session release event (for example, RADIUS Stop Accounting). As part of the session removal process, the BPM knows whether the session has a current SCE mapping (created through a *Login* request) and invokes a corresponding logout request to the SCE for a given subscriber. The system does not include the scope for receiving logout notifications from the SCE. If the BPM receives an unsolicited logout session notification from an SCE, the system ignores it.

Session Removal for Failover Scenario

During an SCE failover, a new *Login* event may be received for an existing session where a SCE session state mapping already exists. Here, The BPM generates a logout event to the original SCE prior to overriding the SCE mapping information delivered through the new *Login* event. The BPM responds to this with a *Login* response that included the current policy profile being applied to that subscriber. The BPM knows the current package assignment, which may not be the default package assignment for that subscriber. (That assignment was set at initial session login.)

Process Session Log out from an SCE

The BPM supports the ability to process the Logout-Response message from the SCE. This does not delay the process of creating the new SCE association within the BPM. The failure to receive the Logout-Response may indicate a complete SCE failure. In this case, the BPM system never receives the response. However, the reason for the failure can be link-related. Here, the SCE is operational and holding subscriber state. This must be resolved.

SCE/BPM Failure

When an SCE fails, its current sessions are redistributed across the other active SCEs within the SCE cluster. This generates new *Login-Pull-Requests* from the SCEs charged with handling the given sessions.

SCE Login Request Queuing

The BPM provides a method of queuing *Login* requests if it receives more requests than can be processed over a given time.

SCE/BPM Resource Controller Failure

The BPM supports Resource Controller failover. This can be achieved using the same IP address for the Primary and Secondary Resource Controller on the network interface that Resource Controller uses for communicating with the SCE.

SCE / BPM Synchronization

In certain circumstances, the BPM performs bulk synchronization with the SCE layer. If the BPM is inserted into a predeployed SCE environment, the BPM extracts the current session state.

- SCE and BPM Synchronization when Resource Controller Fails - The BPM supports bulk synchronization with SCE when a Resource Controller cluster fails, though the Resource Controller clustering approach makes this event unlikely.
- BPM Bulk Synchronization Initiation with SCE - The BPM initiates a bulk synchronization event to extract the current session status of any given SCE. This is selectable for a per SCE instance.

SCE Device Management

- Assign the SCE to Resource Controller - The BPM provides a user-friendly mechanism for assigning SCE devices to Resource Controllers. The BPM supports multiple SCEs (200) from an API perspective.
- SCE Packages - The BPM provides the ability to define and hold a repository of SCE packages that can be applied to relevant SCE IP Sessions.
- Administer the SCE - The BPM permits the SCE administration function to reflect the cluster hierarchy. This permit operations, such as synchronizations to be applied to the cluster as well as individual SCE devices.

Miscellaneous Requirements

- Use of Subscriber IP Address - The Policy Director layer uses the source IP address of the subscriber (passed from the portal over the SOAP interface) to direct the request at the relevant Resource Controller system.
- Application Request as an input for Policy Director Routing - The BPM implements (in the Policy Director Layer) a solution to process the application layer requests to determine the handling of the request (network adaptation only or CAC + network adoption)
- Notifying the User Application - The Policy Director leverages its portal API to signal the success of the action to the SP Portal. This is non-blocking.

Performance Requirements

BPM system does not incur any performance degradation when it supports a dual SAF mode of operation compared to the Single SAF mode of operation. When changes are made to the central Topology Database, they are automatically propagated to impacted SPDF and A-RACF local databases. Some of these changes may be complex, and require careful coordination. For example, if a BRAS moves from one Resource Controller to another (which can happen while the system is running), the central Topology Database initiates a *data move* operation between the source and destination Resource Controller local Topology Databases, and notify the Directors that the impacted BRAS is *in transition*. This state, used during failover, causes corresponding requests to be temporarily rejected. Once the BRAS has moved to the new Resource Controller, the Topology Database notifies Directors of its new location.

Project Modules

This section discusses the following modules:

- Resource Controller
- TDS
- Backend

Resource Controller Protocol Interface Functions

SCE PIF Agent

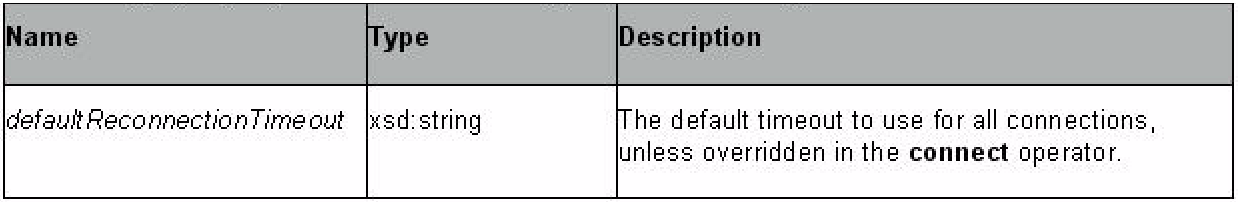
The SCE Protocol Interface Functions (PIF) agent has the following operators:

- *connect*
- *disconnect*
- *loginPullRequest*
- *loginPullResponse*
- *logoutPullRequest*
- *packagePush*
- *connectionDown*

The agent requires a call to the connect operator for a given host before other operators can be used with that host. A persistent connection is maintained by the SCE API, which has an automatic reconnection mode. Loss of connection notification can be detected using the connectionDown trigger operator.

Figure 23 presents the agent properties for use in configuring the SCE PIF agent.

Figure 23. SCE PIF Agent Properties

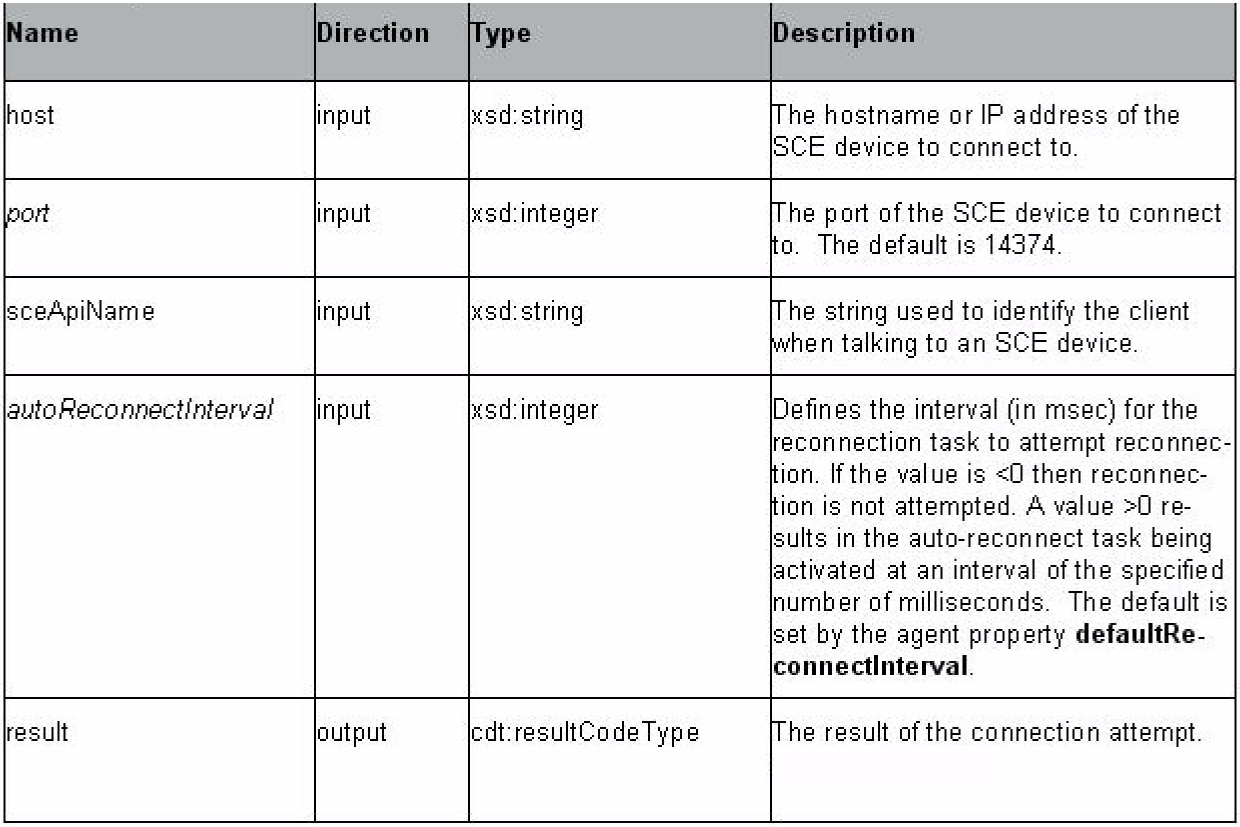


Name	Type	Description
defaultReconnectionTimeout	xsd:string	The default timeout to use for all connections, unless overridden in the **connect** operator.

connect

Before a PIF SCE operation can be made, a connection must explicitly be made with the remote SCE device using the connect PIF. Figure 24 presents the *connect* properties.

Figure 24. Properties - connect

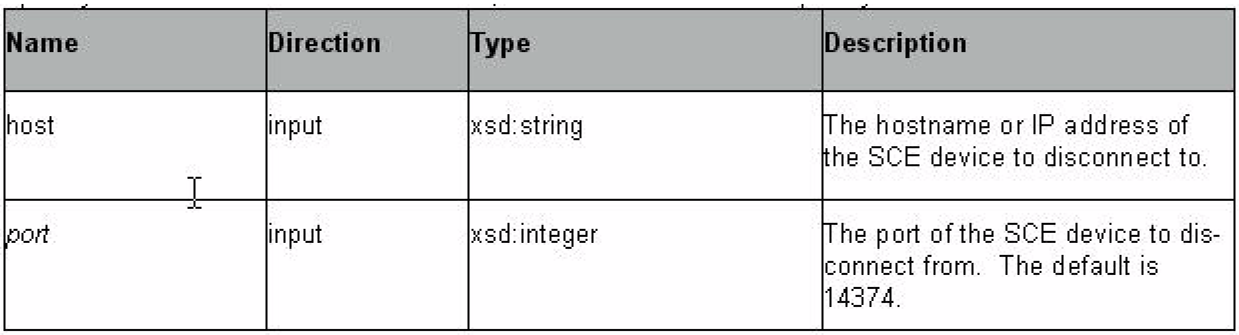


Name	Direction	Type	Description
host	input	xsd:string	The hostname or IP address of the SCE device to connect to.
port	input	xsd:integer	The port of the SCE device to connect to. The default is 14374.
sceApiName	input	xsd:string	The string used to identify the client when talking to an SCE device.
autoReconnectInterval	input	xsd:integer	Defines the interval (in msec) for the reconnection task to attempt reconnection. If the value is <0 then reconnection is not attempted. A value >0 results in the auto-reconnect task being activated at an interval of the specified number of milliseconds. The default is set by the agent property **defaultReconnectInterval**.
result	output	cdt:resultCodeType	The result of the connection attempt.

disconnect

This PIF operation explicitly closes a connection to a remote SCE device. Applications should explicitly close connections. On shutdown, all connections are explicitly closed. Figure 25 presents the *disconnect* properties.

Figure 25. Disconnect Properties

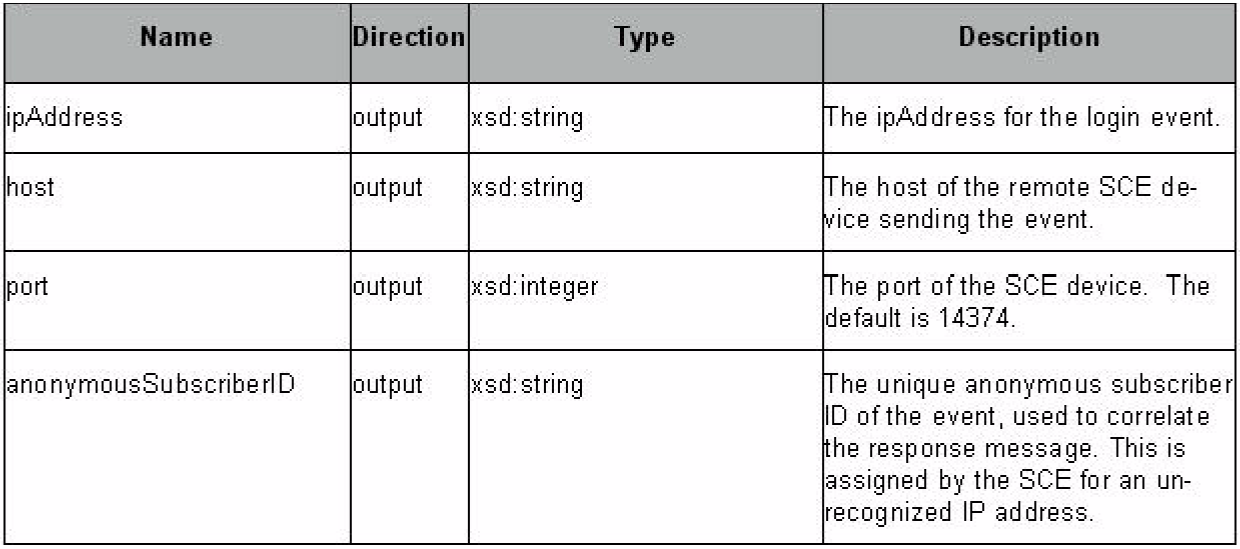


Name	Direction	Type	Description
host	input	xsd:string	The hostname or IP address of the SCE device to disconnect to.
port	input	xsd:integer	The port of the SCE device to disconnect from. The default is 14374.

loginPullRequest

The *loginPullRequest* trigger operator notifies the BPM when the SCE detects a new login event. Figure 26 presents the *loginPullRequest* properties.

Figure 26. Properties - loginPullRequest



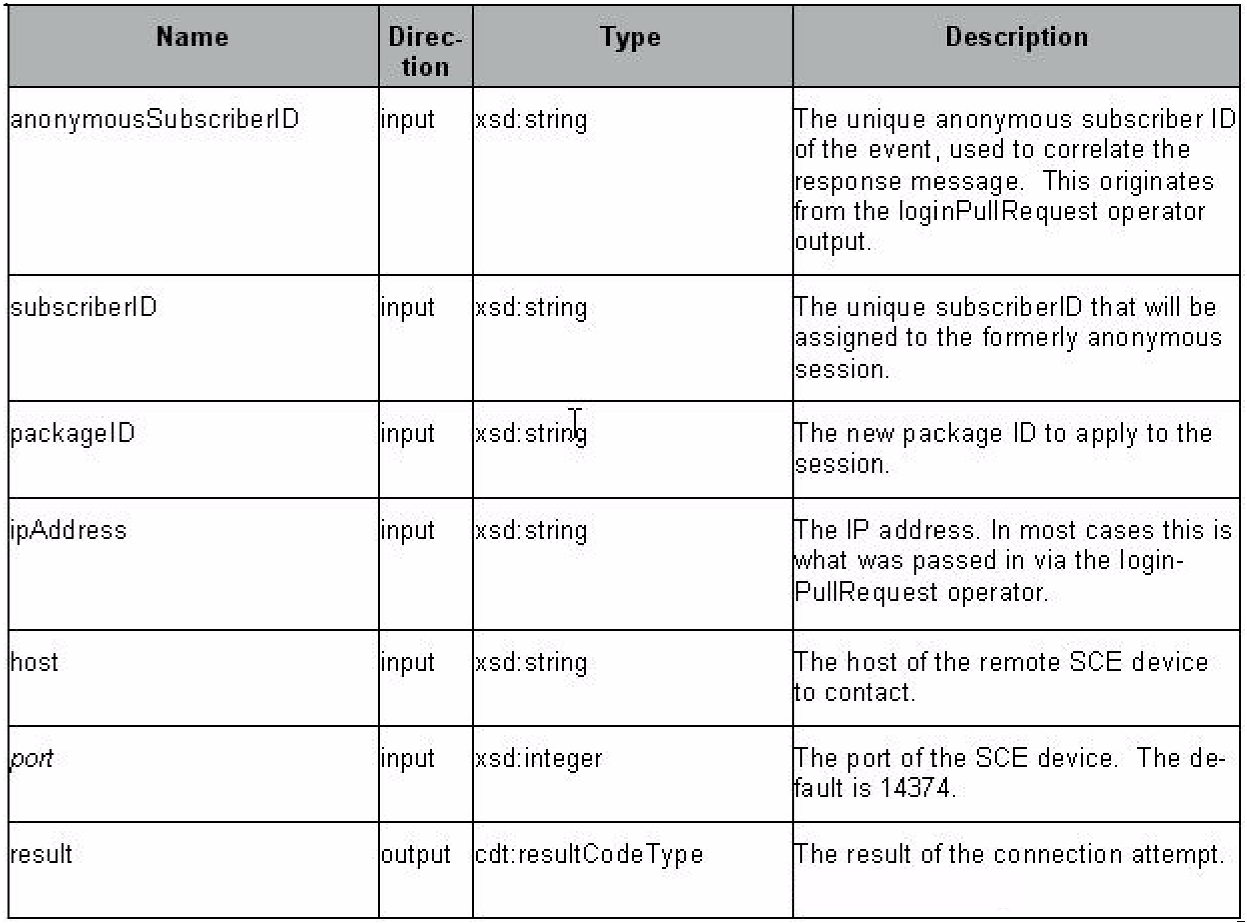
Name	Direction	Type	Description
ipAddress	output	xsd:string	The ipAddress for the login event.
host	output	xsd:string	The host of the remote SCE device sending the event.
port	output	xsd:integer	The port of the SCE device. The default is 14374.
anonymousSubscriberID	output	xsd:string	The unique anonymous subscriber ID of the event, used to correlate the response message. This is assigned by the SCE for an unrecognized IP address.

loginPullResponse

The *loginPullResponse* operator notifies the BPM when the SCE detects a new login event.

Figure 27 presents the *loginPullRequest* properties.

Figure 27. Properties - loginPullRequest

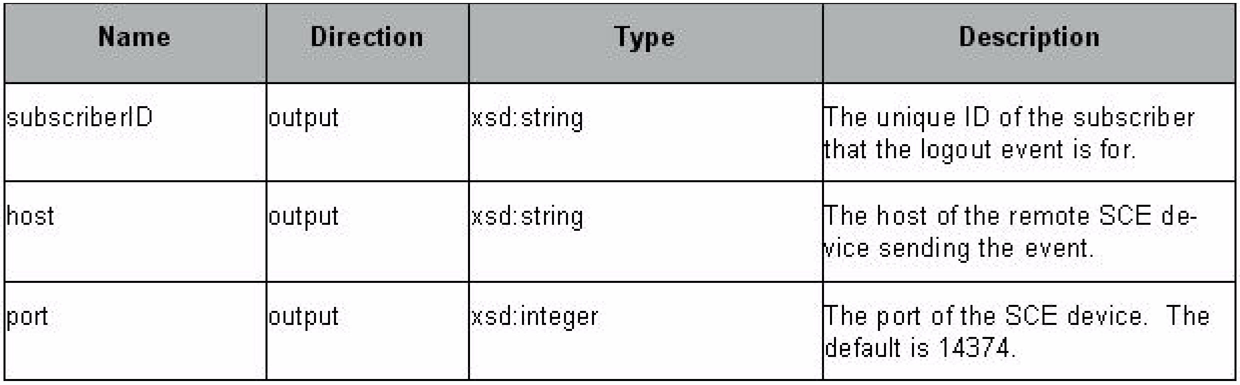


Name	Direc-tion	Type	Description
anonymousSubscriberID	input	xsd:string	The unique anonymous subscriber ID of the event, used to correlate the response message. This originates from the loginPullRequest operator output.
subscriberID	input	xsd:string	The unique subscriberID that will be assigned to the formerly anonymous session.
packageID	input	xsd:string	The new package ID to apply to the session.
ipAddress	input	xsd:string	The IP address. In most cases this is what was passed in via the loginPullRequest operator.
host	input	xsd:string	The host of the remote SCE device to contact.
port	input	xsd:integer	The port of the SCE device. The default is 14374.
result	output	cdt:resultCodeType	The result of the connection attempt.

logoutPullRequest

The *logoutPullRequest* operator is a trigger operator that notifies the BPM when the SCE detects a subscriber logout event. Figure 28 presents the *logoutPullRequest* properties.

Figure 28. Properties - logoutPullRequest

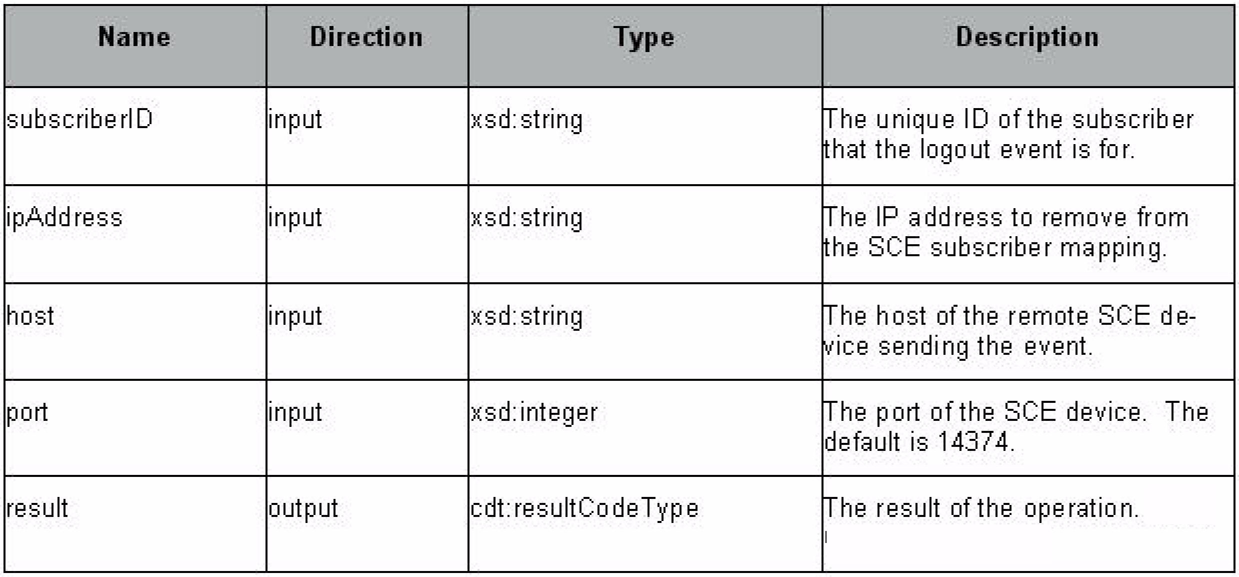


Name	Direction	Type	Description
subscriberID	output	xsd:string	The unique ID of the subscriber that the logout event is for.
host	output	xsd:string	The host of the remote SCE device sending the event.
port	output	xsd:integer	The port of the SCE device. The default is 14374.

logout

The logout operator is used to remove a specific subscriber's network mapping. Figure 29 presents the *logout* properties.

Figure 29. Properties - logout

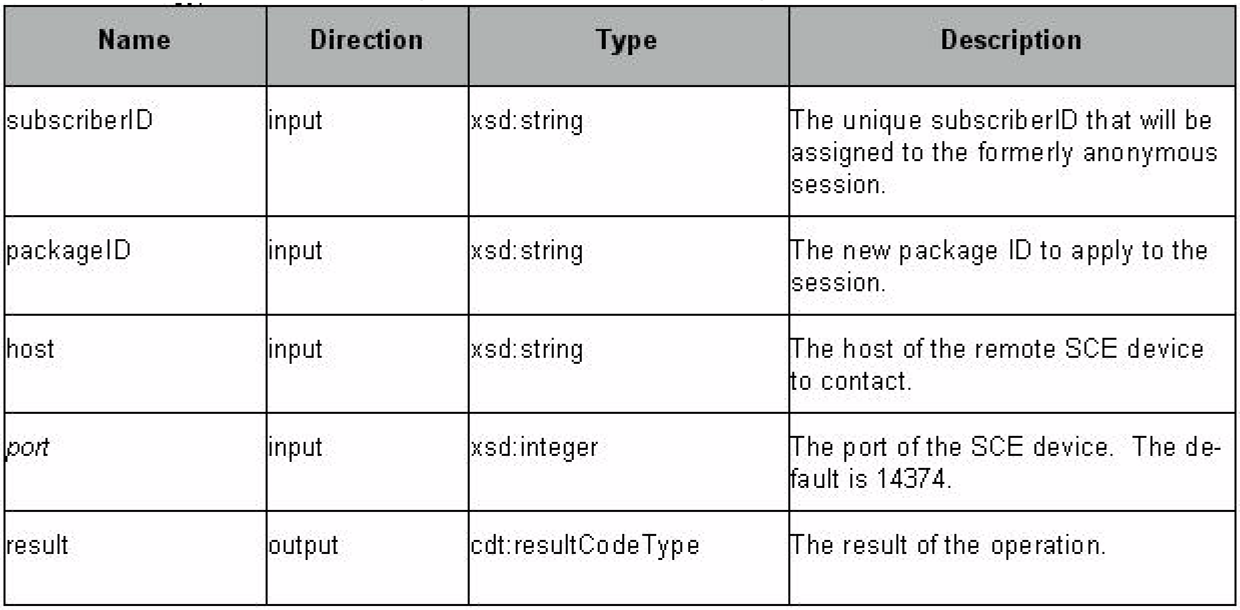


Name	Direction	Type	Description
subscriberID	input	xsd:string	The unique ID of the subscriber that the logout event is for.
ipAddress	input	xsd:string	The IP address to remove from the SCE subscriber mapping.
host	input	xsd:string	The host of the remote SCE device sending the event.
port	input	xsd:integer	The port of the SCE device. The default is 14374.
result	output	cdt:resultCodeType	The result of the operation.

packagePush

The packagePush operator applies a given package to a session identified by a unique subscriber ID (SCE terminology). Figure 30 presents the *packagePush* properties.

Figure 30. Properties - packagePush

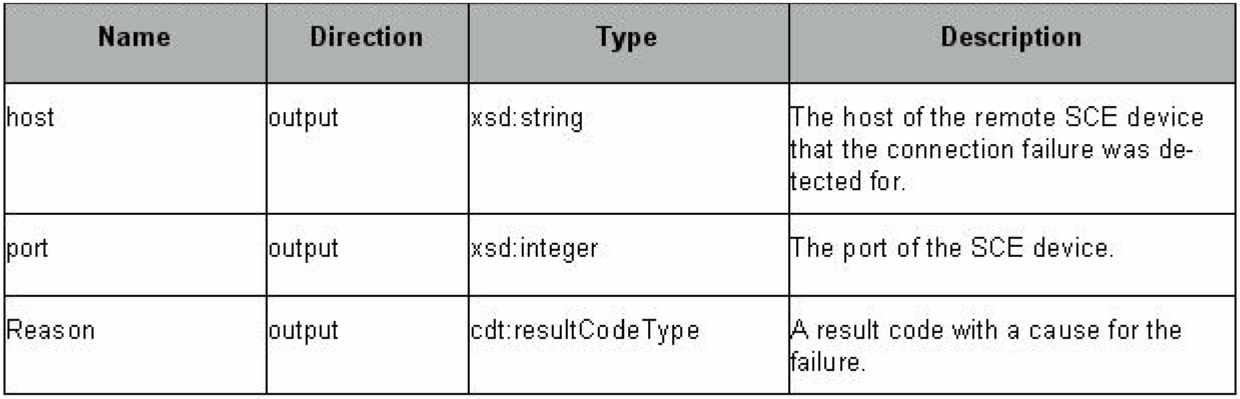


Name	Direction	Type	Description
subscriberID	input	xsd:string	The unique subscriberID that will be assigned to the formerly anonymous session.
packageID	input	xsd:string	The new package ID to apply to the session.
host	input	xsd:string	The host of the remote SCE device to contact.
port	input	xsd:integer	The port of the SCE device. The default is 14374.
result	output	cdt:resultCodeType	The result of the operation.

connectionDown

The *connectionDown* operator is a trigger operator that can be used to detect a connection failure to a specific host. This is determined by the underlying SCE API. Figure 31 presents the *connectionDown* properties.

Figure 31. Properties - connectionDown



Name	Direction	Type	Description
host	output	xsd:string	The host of the remote SCE device that the connection failure was detected for.
port	output	xsd:integer	The port of the SCE device.
Reason	output	cdt:resultCodeType	A result code with a cause for the failure.

Resource Controller Device Adapter Function

The SCE Device Adapter Functions (DAFs) handle both northbound and southbound actions. Two DAFs implement the required functionality: *login* and *applyProfile*.

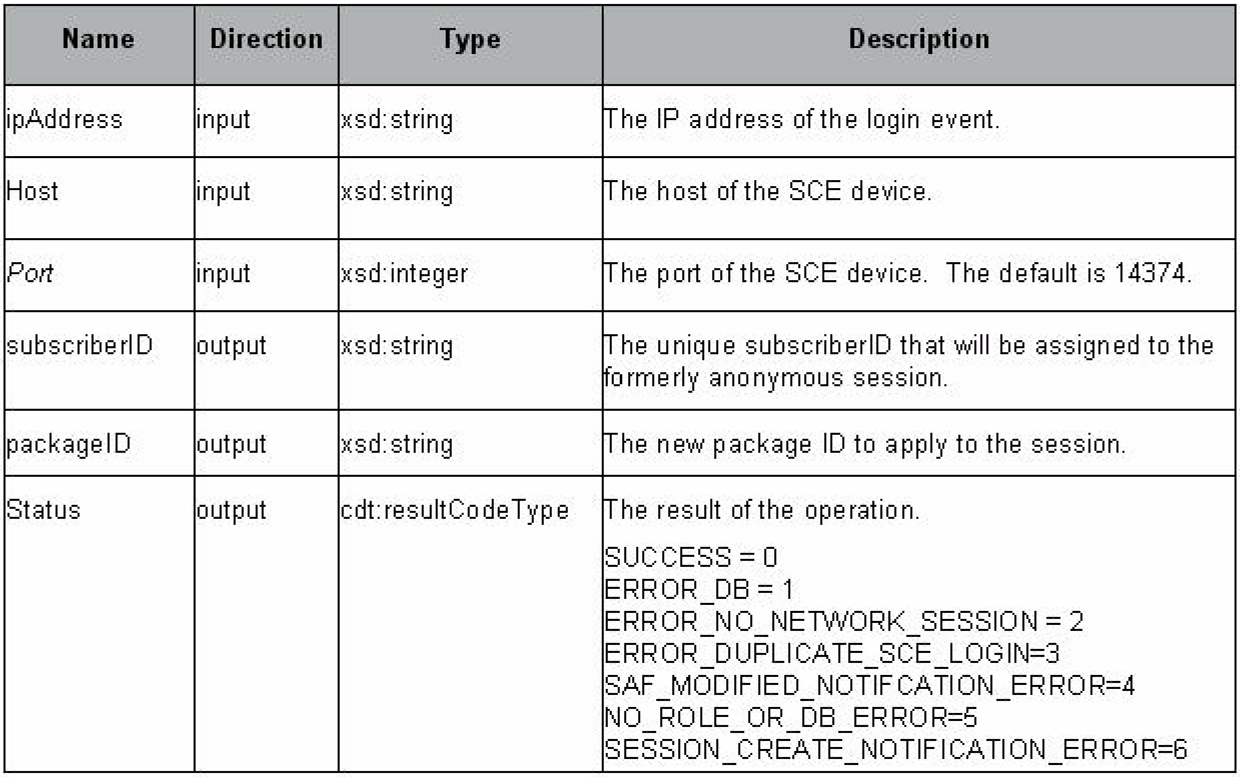
SCE login DAF

The SCE login DAF flow is called by the SCE *loginPull* PIF. Figure 32 presents the l*oginPull* properties.

Interface

Figure 32 presents the *loginPull* properties.

Figure 32. Properties - loginPull



Name	Direction	Type	Description
ipAddress	input	xsd:string	The IP address of the login event.
Host	input	xsd:string	The host of the SCE device.
Port	input	xsd:integer	The port of the SCE device. The default is 14374.
subscriberID	output	xsd:string	The unique subscriberID that will be assigned to the formerly anonymous session.
packageID	output	xsd:string	The new package ID to apply to the session.

Status	output	cdt:resultCodeType	The result of the operation.
			SUCCESS = 0
			ERROR_DB = 1
			ERROR_NO_NETWORK_SESSION = 2
			ERROR_DUPLICATE_SCE_LOGIN=3
			SAF_MODIFIED_NOTIFCATION_ERROR=4
			NO_ROLE_OR_DB_ERROR=5
			SESSION_CREATE_NOTIFICATION_ERROR=6

Operation

The SCE login DAF flow is called by the SCE loginPull PIF.

1. Receive trigger message with an IP Address and SCE address.
2. Look up network session in SSF using IP Address as the attribute value for *public:userIpAddress*.
   a. If session does not exist, return ERROR_NO_NETWORK_SESSION to caller (PIF does not send response).
   b. Else continue to step 3.
3. Check if a device session for the SCE role already exists.
   a. If a device session exists for the SCE role with a different IP address than the input, change the device session to point to the new SCE address and call the *pif_sce/logout* flow to remove the subscriber mapping from the old SCE.
   b. If a device session exists with the same IP address, log: *Duplicate SCE Login received for <IP Address> from <host>*, go to step 5.
   c. Otherwise, create new a device session using SCE host and create a *public:networkIpAddress* key from input IP address.
4. Call *SAF/sessionModified* to notify the SM Application of a modification to the network session (in this case, a device session modification or creation).

5. Look up the current profile (with actions) using the SSF Agent *getCurrentProfile* operator, passing in the device instance ID.
6. Look up the instructions for the SCE device action (*packageID*) referenced in the returned profile in the above step.
7. Return the current profile and the unique identifier (network session ID) retrieved the network session to the caller.

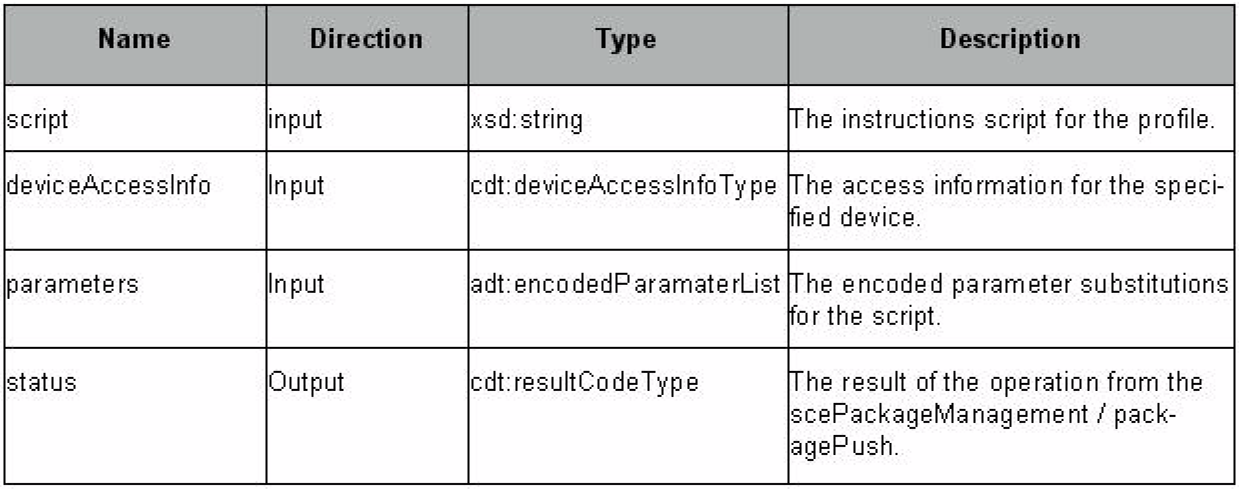
SCE applyProfile DAF

The SCE *applyProfile* DAF is called by the PMF. It retrieves the proper package name to send to the SCE device for the given profile.

Interface

These are encoded as part the attributes of the standard DAF interface. Figure 33 presents the DAF interface properties.

Figure 33. DAF Interface Properties



Name	Direction	Type	Description
script	input	xsd:string	The instructions script for the profile.
deviceAccessInfo	Input	cdt:deviceAccessInfoType	The access information for the specified device.
parameters	Input	adt:encodedParamaterList	The encoded parameter substitutions for the script.
status	Output	cdt:resultCodeType	The result of the operation from the scePackageManagement / packagePush.

Operation

The SCE applyProfile DAF behaves as follows:

1. Decode input parameters from standard DAF interface attributes parameter.
2. Pull out the host and port name from the encoded *deviceAccessInfo* input.
3. Invoke the SCE packagePush PIF using host (decode from *deviceAccessInfo* input), port (decoded from *deviceAccessInfo*), packageID (using the script input), and *subscriberID* (using the *networkSessionID* input).
4. Report PIF status to caller.

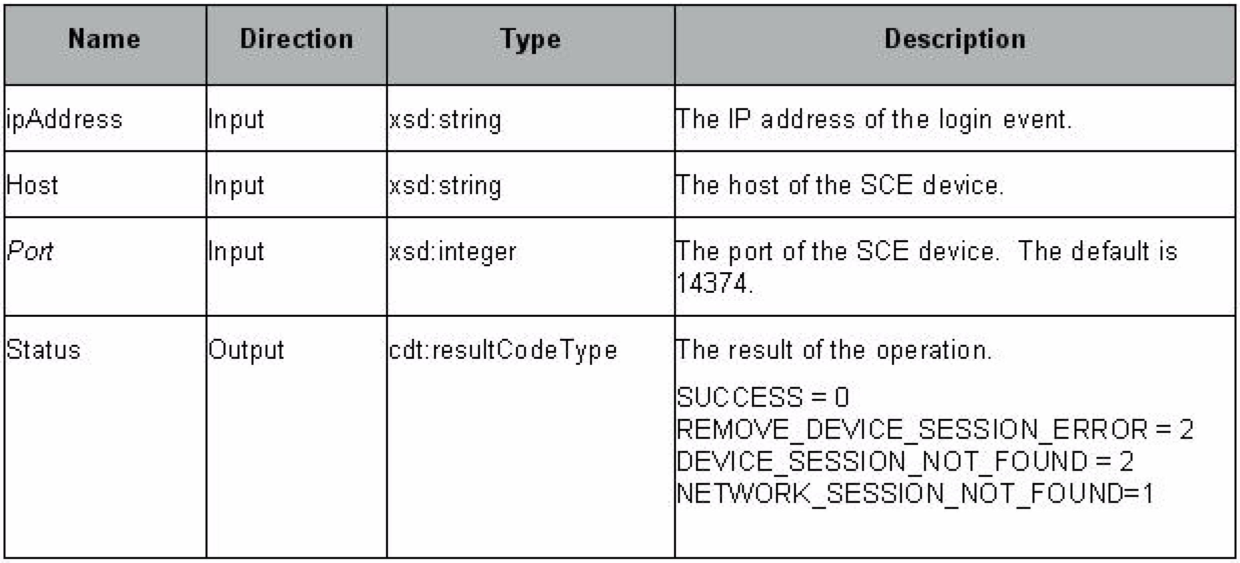
SCE logout DAF

The SCE login DAF flow is called by the SCE *logoutPull* PIF.

Interface

Figure 34 presents the *logoutPull* properties.

Figure 34. Properties - logoutPull



Name	Direction	Type	Description
ipAddress	Input	xsd:string	The IP address of the login event.
Host	Input	xsd:string	The host of the SCE device.
Port	Input	xsd:integer	The port of the SCE device. The default is 14374.

Status	Output	cdt:resultCodeType	The result of the operation.
			SUCCESS = 0
			REMOVE_DEVICE_SESSION_ERROR = 2
			DEVICE_SESSION_NOT_FOUND = 2
			NETWORK_SESSION_NOT_FOUND=1

Operation

The SCE logout DAF flow is be called by the SCE logoutPull PIF.

1. Receive trigger message with an IP Address and SCE address.
2. Look up network session in SSF using IP Address as the attribute value for *public:userIpAddress.*
    a. If network session does not exist, return ERROR_NO_NETWORK_SESSION to caller. (PIF does not send response.)
    b. If network session exists, continue to Step 5.
3. Find the Device Session.
    a. If device session does not exist, return ERROR_NO_DEVICE_SESSION to caller. (PIF does not send response.)
    b. If device session exists, continue to step 8.
4. Remove the device session.
5. Return the status to the caller.

TDS Data Model Modifications

To handle the mapping of a SCE instance to a Resource Controller, Resource Controllers are stored within the domain realm, while device instances (SCE devices) are stored in the network realm. The mapping to the Resource Controller is maintained by adding information to the domain realm.

The domain realm is extended with the *deviceInstance* resource. This resource is associated with a node representing the active and standby Resource Controllers at provisioning time. In this release, the *deviceInstance* resource in the domain resource has no direct connection to the *deviceInstance* in the network realm. Devices may be duplicated in both realms.

The *deviceInstance* resource in the domain contains only an *id*, which should be identical to the *id* in for the corresponding *deviceInstance* object in the network realm.

The domain realm is stored on the TDS, and is accessed by the Resource Controller initially to determine the SCE devices that it is in charge of, using the *tds_tsf* services. Resource Controllers are notified of SCE reassignment by a publicly available service.

Note: The TDS maintains a mapping of devices to Resource Controller cluster, not of device to realm and realm to device.

Backend

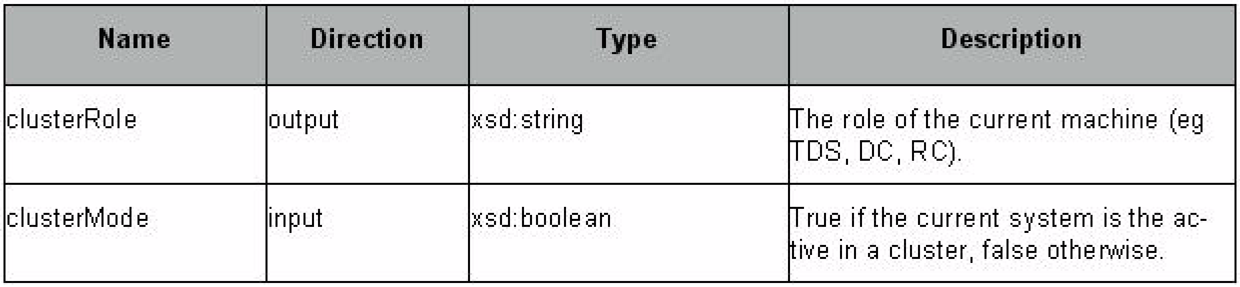
Service Deployed Hook

The existing hooks system creates persistent connection to an SCE device at deployment time. This is done by adding a generalized system of allowing one flow per service to be called immediately after that service has been deployed. Service creators can create a flow called *__deploy__* to be invoked by a hook using Vista. This flow is triggered upon successful deployment of the service. The *__deploy__* flow must be triggered via the socket agent and has a well-defined interface.

Generic __deploy__ Interface

Figure 35 presents the *Generic_deploy_Interface* properties.

Figure 35. Properties - Generic_deploy_Interface



Name	Direction	Type	Description
clusterRole	output	xsd:string	The role of the current machine (eg TDS, DC, RC).
clusterMode	input	xsd:boolean	True if the current system is the active in a cluster, false otherwise.

Procedure: SCE PIF __deploy__ Operation

Use the existing hooks system to create a persistent connection to an SCE device at deployment time.

1. Query TSF on TDS realm using *tds_tsf* service to find all SCE devices associated with Resource Controller

2. For each SCE device found, Invoke SCE PIF/connect flow with SCE device address to create a connection.

Appendix A - Glossary

This appendix contains abbreviations, acronyms, terms, and their definitions.

Table 4-1. Terms and Definitions.

Term	Definition
A	
Accounting Log Function	ALF. The Accounting Log Function records entrance parameters, internal decisions, and exit responses.
ACF	Admission Control Function. The ACF provides the core logic for performing admission control. It is programmed with a set of policies that define admission control behavior.
Action	An action is an operational category for changing, or inquiring about, a network element.
Active BPM	In a pair of BPMs, the active BPM processes requests. A standby BPM constantly monitors the health of the active BPM. If the active BPM is not viable, the standby BPM becomes the active BPM.
Admission Control Function	ACF. The ACF provides the core logic for performing admission control. It is programmed with a set of policies that define admission control behavior.
Agent	An internal BPM component that interacts with a device. The designer creates the agent and configures it to interact with a specific device by indicating the device type, IP address, and port number. The designer then assigns the agent to perform service functions.
Agent Configuration	Agent information that comprises a specific agent type instance. For example, a RADIUS agent configuration contains appropriate IP address, port, and shared secret values for a RADIUS agent type.
Agent Function	The service designer uses the BPDS to drag and drop an agent function into a flow in the BPDS. An agent, interacting with a device, performs the actual operation.
Agent Instance	A running instance of an agent type.
Agent Package	Software that allows agents to interact with a particular device type. For example, a RADIUS agent package contains software that allows the creation of agents that interact with specific RADIUS devices.
Agent Type	The agent type describes a particular type of agent that you can load onto the system. You select the agent type when you create the agent instance.
AI	Application Interface. The underlying frameworks use Application Interfaces to notify the Application of network events. The Cisco framework provides these interfaces.
Alarm Notification Function	ANF. The Alarm Notification Function issues SNMP traps to alert external systems of aberrant behavior in the BPM.

Table 4-1. Terms and Definitions.

Term	Definition
ALF	Accounting Log Function. The Accounting Log Function records entrance parameters, internal decisions, and exit responses.
ANF	Alarm Notification Function. The Alarm Notification Function (ANF) issues SNMP traps to alert external systems of aberrant behavior in the BPM.
API	Application Program Interface. An API is a set of routines, protocols, and tools for building software applications. An API makes it easier to develop a program by providing the required building blocks. A programmer puts the blocks together.
Application	A service that maps business models and operational procedures directly into IP services, executable by their customers, for example, video on demand or automatic backup. See also Service.
Application Interface	AI. The underlying frameworks use Application Interfaces to notify the Application of network events. The Cisco framework provides these interfaces.
Application Program Interface	API. An API is a set of routines, protocols, and tools for building software applications. An API makes it easier to develop a program by providing the required building blocks. A programmer puts the blocks together.
Application Service Provider	ASP. An ASP is a business that provides computer-based services to customers over a network.
ASP	Application Service Provider. An ASP is a business that provides computer-based services to customers over a network.
Asynchronous Transfer Mode	ATM. Asynchronous Transfer Mode is a network technology based on transferring data in cells or packets of a fixed size.
ATM	Asynchronous Transfer Mode. ATM is a network technology based on transferring data in cells or packets of a fixed size.
Attribute	An attribute is a datum about a network session or a device session. Attributes contain a name and value and a distinguishing namespace. In the BPDS Object Manager tool, a simple type with a default value. An object can have several attributes.
B	
Backend	Software that runs on the BPM. It comprises the controller, engine, agent host, activation daemon, and scheduler processes; synonymous with BPM.
BGP	Border Gateway Protocol. An exterior gateway routing protocol that enables groups of routers to share routing information to establish efficient, loop-free routes. BGP is commonly used within and between ISPs.
Border Gateway Protocol	BGP. An exterior gateway routing protocol that enables groups of routers to share routing information to establish efficient, loop-free routes. BGP is commonly used within and between ISPs.
BPDS	Broadband Policy Design Studio. The BPDS is a graphical user interface to the BPM. The BPDS includes a service design feature.
BPM	Broadband Policy Manager. The BPM is a product suite used by service providers to create and deploy advanced services on broadband networks. A BPM system can be configured as a Director, Domain Controller, Resource Controller, or Topology Database Server.

Table 4-1. Terms and Definitions.

Term	Definition
BPS	Broadband Policy Studio. The BPS is a graphical user interface, similar to the BPDS. The BPS does not include the service design feature.
BRAS	Broadband Remote Access Server. A BRAS device routes traffic to and from the digital subscriber line access multiplexers on an ISP network.
Broadband Policy Design Studio	BPDS. The BPDS is a graphical user interface, similar to the BPS. The BPDS includes a service design feature.
Broadband Policy Manager	BPM. The BPM is a product suite used by service providers to create and deploy advanced services on broadband networks.
Broadband Policy Studio	BPS. The BPS is a graphical user interface, similar to the BPDS. The BPS does not include the service design feature.
Broadband Remote Access Server	BRAS. A BRAS device routes traffic to and from the digital subscriber line access multiplexers (DSLAM) on an ISP network.
C	
CAC	Capacity Admission Control. CAC monitors, controls, and enforces the use of network resources and services with policy-based management over broadband access and MPLS core networks.
Capacity Admission Control	CAC. CAC monitors, controls, and enforces the use of network resources and services with policy-based management over broadband access and MPLS core networks.
Cisco Network Registrar	CNR. The CNR is a full-featured DNS/DHCP system that provides scalable naming and addressing services for service provider and enterprise networks.
Class of Service	CoS. This is a traffic prioritization scheme that enables more predictable traffic delivery, based on application requirements.
Classless Inter-Domain Routing	CIDR. This IP addressing scheme addresses the size of routing tables and makes more IP addresses available within organizations. CIDR is also called supernetting.
CIDR	Classless Inter-Domain Routing. This IP addressing scheme addresses the size of routing tables and makes more IP addresses available within organizations. CIDR is also called supernetting.
Client	This is a generic term that denotes the BPM BPDS application.
CoS	Class of Service. This is a traffic prioritization scheme that enables more predictable traffic delivery, based on application requirements.
CPE	customer premises equipment. This is communications equipment that resides on the customer premises. It is owned or leased by the customer.
CLI	command line interface. This is a user interface common to computers. The user enters a command. The computer acts on the command.
Cluster	A pair of cooperating and redundant BPMs.
CNR	Cisco Network Registrar. The CNR is a full-featured DNS/DHCP system that provides scalable naming and addressing services for service provider and enterprise networks.
Command Line Interface	CLI. This is a user interface common to computers. The user enters a command. The computer acts on the command.

Table 4-1. Terms and Definitions.

Term	Definition
Component	An object comprising data and code. A component provides a well-specified set of publicly available services. All devices, services, and applications on a network are components.
Configuration	Information necessary to construct an instance of a type (agent, service).
Controller	A software element that runs on the BPM and controls various elements of the backend. Usually only one controller exists per backend; therefore, from the BPDS perspective, the controller is the backend.
Customer Premises Equipment	CPE. This is communications equipment that resides on the customer premises. It is owned or leased by the customer.
D	
DAF	Device Adapter Function. A DAF translates between the protocol and device type-specific events of at the PIF layer and the abstract application events at the Application layer. A DAF can be assigned to multiple device types and multiple DAFs can be assigned to one device.
Deep Packet Inspection Protocol	DPI. This is network packet filtering that examines packet *data*, searching for nonprotocol compliance or predefined criteria, to decide if the packet can pass. This is in contrast to shallow packet inspection (called packet inspection), which checks only the packet *header*.
Device	Any piece of software or hardware connected to a network. RADIUS servers, routers, billing systems, accounting systems, and video servers are devices. An agent communicates with a device.
Device Access	A device access is data about accessing a device instance. Most devices require authentication before any device action can occur. The device access contains this authentication data and other related data. Each device instance has one device access per management protocol.
Device Action	A device action is the implementation of an action for a given device type. That is, it is the actual set of instructions necessary to change the functioning of the device instance.
Device Adapter Function	DAF.A DAF translates between the protocol and device type-specific events of at the PIF layer and the abstract application events at the Application layer. A DAF can be assigned to multiple device types and multiple DAFs can be assigned to one device.
Device Adapter Function Flow	A Flow that handles a protocol event for a specific device type.
Device Handler Dispatch Service	DHDS. DHDS provides routing services for PIFs and Applications requesting invocation of DAF operations.
Device Instance	A device instance is a device type in use in the network. For example, a Cisco 10K device at IP address 128.148.176.10. Device instances are grouped according to roles.

Table 4-1. Terms and Definitions.

Term	Definition
Device Rule	A device rule is a provisioned list of steps that apply a policy to a device. A device rule consists of a set of instructions that the BPM sends to the device to apply the given policy. Device rules can retrieve information from connected devices. Preconfigured device rules are useful for configuring a new BPM system. See also Device Type and Policy Rule.
Device Session	A device session contains data about a device instance used by a network session. For example, information about the bras would be encoded in a device session.
Device Type	A device type is a vendor's network element hardware. Device types are grouped according to roles and are based on device attributes, such as vendor, model, hardware version, and software version. See also Device Rule.
DHDS	Device Handler Dispatch Service. DHDS provides routing services for PIFs and applications requesting invocation of DAF operations.
Digital Subscriber Line	DSL. DSL technologies use sophisticated modulation schemes to pack data onto copper wires.
Digital Subscriber Line Access Multiplexer	DSLAM. This mechanism links customer DSL connections to a single high-speed ATM line.
Director	A Director is one or more stateless installations that takes requests and routes them to appropriate Resource Controllers, to handle the specific incoming requests.
Director Realm	The Director Realm stores information required by Director systems, including information about network devices (such as BRAS devices). The information specifies the Resource Controller responsible for each device and the IP address pools each device handles. A Director uses this information to forward an incoming request to the correct Resource Controller. The Topology Database Server maintains the Director Realm, and the server distributes its updates to each Director when updates occur.
Domain	One or more cooperating Broadband Policy Managers (BPMs) managed by a single domain repository.
Domain Controller	The Domain Controller is a standalone system responsible for domain management, including application deployment, configuration, and health for all systems in the domain. Only one Domain Controller exists per domain.
Domain Data	Data maintained about the elements in a domain; for example, controller host and port configuration, database host and port information, agent and service configuration and deployment information.
Domain Realm	The Domain Realm maintains application level information about the physical network topology. The nodes in the topology represent Director and Resource Controller systems. The Topology Database Server uses the Domain Realm to understand the system topology. Links represent connectivity between cluster pairs. Resources represent interfaces on the component systems, system health, cluster information, and system configuration.

Table 4-1. Terms and Definitions.

Term	Definition
Domain Repository	The master database that contains configuration information for each domain element.
DPI	Deep Packet Inspection Protocol. This is network packet filtering that examines packet *data*, searching for nonprotocol compliance or predefined criteria, to decide if the packet can pass. This is in contrast to shallow packet inspection (called packet inspection), which checks only the packet *header*.
DSL	Digital Subscriber Line. DSL technologies use sophisticated modulation schemes to pack data onto copper wires.
DSLAM	Digital Subscriber Line Access Multiplexer. This mechanism links customer DSL connections to a single high-speed ATM line.
E	
Element	An object with the BPM: package; agent configuration; service instance; shared object.
Enumeration	In the BPDS Object manager tool, enumeration is contained within a simple type.
Ethernet	The Ethernet is a large and diverse family of frame-based computer networking technologies for local area networks (LANs). It defines a number of wiring and signaling standards for the physical layer, two means of network access at the Media Access Control (MAC)/Data Link Layer, and a common addressing format. Ethernet has been standardized as IEEE 802.3.
ETSI	European Telecommunications Standards Institute. ETSI is an independent, non-profit organization, whose mission is to produce telecommunications standards for today and for the future.
European Telecommunications Standards Institute	ETSI. ETSI is an independent, non-profit organization, whose mission is to produce telecommunications standards for today and for the future.
F	
Field Replaceable Unit	FRU. An FRU represents an element (e.g., entire system, BPDS client software, agent) within the Broadband Policy Managers (BPM) that has a version associated with it. A FRU is a subset of an element.
Flow	The movement of data or control between agents. It is a collection of one or more operators and zero or more routes. The designer uses flows to define services and applications.
FRU	Field Replaceable Unit. An FRU represents an element (e.g., entire system, BPDS client software, agent) within the Broadband Policy Managers (BPM) that has a version associated with it. A FRU is a subset of an element.
Function	The element that performs an operation, based on inputs and returns the results of the operation via its outputs. The designer drags and drops a function into a flow in the BPDS. An agent, interacting with a device, performs the actual operation.
G	

Table 4-1. Terms and Definitions.

Term	Definition
Graphical User Interface	GUI. A program interface that takes advantage of the computer's graphics capabilities to make the program easier to use. For the BPM, the GUI is the BPDS.
GUI	Graphical User Interface. A program interface that takes advantage of the computer's graphics capabilities to make the program easier to use. For the BPM, the GUI is the BPDS.
H	
Handler	A handler enables flow of control between the PIF, DAF, and SMF interfaces. It includes details about the appropriate service flow to call under specific conditions.
Handler Flow	A Handler Flow normalizes protocol-specific parameters before forwarding them to an application. An application can indirectly invoke a Handler Flow using the DHDS.
Head Version	The latest version of an element.
Hypertext Preprocessor	PHP. PHP is an open source, server-side, HTML embedded scripting language used to create dynamic Web pages.
I	
Implementation	An instruction set for executing a specification.
Instance	An executing type (agent, service), created from a specification, implementation, and configuration. An agent instance is a specific implementation of that agent type.
Interface	A collection of functions.
Internet Service Provider	ISP. An ISP is a company that provides access to the Internet. For a monthly fee, the company provides a software package, username, password and access phone number. In addition to serving individuals, ISPs also serve large companies, providing a direct connection from the company network to the Internet.
IP address	The address that identifies a computer. The IP address format is a 32-bit numeric address written as four numbers (0 to 255) separated by periods.
ISP	Internet Service Provider. An ISP is a company that provides access to the Internet. For a monthly fee, the company provides a software package, username, password and access phone number. In addition to serving individuals, ISPs also serve large companies, providing a direct connection from the company network to the Internet.
J	
K	
Key	A key is an identifier used in conjunction with network sessions.
L	
LAN	Local Area Network. A LAN is computer network that spans a relatively small area. Most LANs are confined to a single building or group of buildings. A LAN connect workstations and personal computers. This allows users to share devices and data and communicate via email.
L2TP	Layer Two Tunneling Protocol. L2TP is an extension to the PPP protocol that enables ISPs to operate VPNs.

Table 4-1. Terms and Definitions.

Term	Definition
Layer Two Tunneling Protocol	L2TP. L2TP is an extension to the PPP protocol that enables ISPs to operate VPNs.
Link	A link is a line or channel over which data is transmitted.
Local Area Network	LAN. A LAN is computer network that spans a relatively small area. Most LANs are confined to a single building or group of buildings. A LAN connect workstations and personal computers. This allows users to share devices and data and communicate via email.
M	
Management Protocol	A management protocol is the mechanism for managing a network element. Common management protocols are RADIUS and SNMP.
Metadata	In the BPDS, this is the data structure. A customer can import metadata to invoke a structure for his or her database.
MPLS	Multiprotocol Label Switching. MPLS integrates Layer 2 network link information into Layer 3 within an autonomous system or ISP. It improves IP-packet exchange and allows operators to divert and route traffic around link failures, congestion, and bottlenecks.
Multiprotocol Label Switching	MPLS. MPLS integrates Layer 2 network link information into Layer 3 within an autonomous system or ISP. It improves IP-packet exchange and allows operators to divert and route traffic around link failures, congestion, and bottlenecks.
N	
N + 1 Redundancy	The ability for service engines to use one service engine as a backup.
NAF	Network Adaptation Function. NAF. The NAF dynamically resizes network links and queue sizes, based on the ability of the underlying network to adapt after a request from the ACF.
Namespace	A namespace helps distinguish two or more values that otherwise would conflict with each other.
NAS	Network Attached Storage. A NAS device is a server dedicated to file sharing, allowing more hard disk storage space to be added to a network that already utilizes servers without shutting them down for maintenance and upgrades. A NAS device can exist anywhere in a LAN and can be made up of multiple networked NAS devices.
NAV	Network Admin view. In the BPS and BPDS graphical user interfaces to the BPM, this is the network view where you can perform administration tasks.
Network	A network is a group of two or more computer systems linked together. Local-area networks (LANs), wide-area networks (WANs), and metropolitan-area networks MANs are typical networks.
Network Adaptation Function	NAF. The NA) dynamically resizes network links and queue sizes, based on the ability of the underlying network to adapt after a request from the ACF.
Network Admin View	NAV. In the BPS and BPDS graphical user interfaces to the BPM, this is the network view where you can perform administration tasks.

Table 4-1. Terms and Definitions.

Term	Definition
Network Attached Storage	NAS. A NAS device is a server dedicated to file sharing, allowing more hard disk storage space to be added to a network that already utilizes servers without shutting them down for maintenance and upgrades. A NAS device can exist anywhere in a LAN and can be made up of multiple networked NAS devices.
Network Event	A network event is a set of install and uninstall rules, contained within a profile, that are performed in sequence.
Network Manager	NM. The NM product provides a framework for controlling and querying the element configurations in the broadband network.
Network Policy	A network policy is a device rule entry. The device rule contains commands to configure a network device to apply a network policy. See also Device Rule, Policy Rule.
Network Realm	The Network Realm stores specific network adaptation information, such as the devices active on a particular Resource Controller, profiles, and handlers. The Network Realm is centrally provisioned on the Topology Database Server, and it is distributed to all Resource Controllers.
Network Session	A network session represents a single point-to-point connection in the network, for example, a VoIP call.
Network Storage Function	NSF. The Network Storage Function provides access to the Network Information Model.
NM	Network Manager. The NM product provides a framework for controlling and querying the element configurations in the broadband network.
Node	In networks, a processing location. A node can be a computer or some other device, such as a printer. Every node has a unique network address, sometimes called a Data Link Control (DLC) address or Media Access Control (MAC) address.
NSF	Network Storage Function. The NSF provides access to the Network Information Model.
O	
Object	An agent, controller, function, service, switch, or service within the Broadband Policy Manager (BPM).
Object Dependency	An exact object type, for example a Cisco 2500 router agent, that a service depends on. The service designer adds the object type to the dependency list of the service. All Interfaces supported by the object type are then available for use with the service.
Object Type	In the BPDS, an object type is defined with attributes. It can own contain, and associate with other object types.
OC	Orchestration Controller. That portion of the Broadband Policy Managers (BPM) that controls processes such as username and password authentication.
Operation and Support System	OSS. OSS refers to a suite of programs that enable an enterprise to monitor, analyze, and manage a network system. The term originally referred to a management system that controlled telephone and computer networks. It now applies to the business world to mean a system that supports network operations.

Table 4-1. Terms and Definitions.

Term	Definition
Operator	A representation of actions to be undertaken on a system networked to a Broadband Policy Managers (BPM).
Orchestration Controller	OC. That portion of the Broadband Policy Managers (BPM) that controls processes such as username and password authentication.
Orchestration Network	The process for handling service calls over a network. It defines the flow of control and information between work units.
OSS	Operation and Support Systems. OSS refers to a suite of programs that enable an enterprise to monitor, analyze, and manage a network system.
P	
Pad	A collection of pins on an operator. This appears as a box along the edge of an operator.
Path Computation Function	PCF. The PCF determines the path through the topology for any given end-to-end session, as requested by the ACF.
PCF	Path Computation Function. The PCF determines the path through the topology for any given end-to-end session, as requested by the ACF.
PDP	Policy Decision Point. The PDP is a component of policy-based management. When a user tries to access a file or other resource on a system using policy-based access management, the PDP decides whether or not to authorize the user based on user attributes.
PE	Policy Engine. The software that stores and manages user profile information, subscriber access records, policy rules; also known as the policy database.
PEP	Policy Enforcement Point. The PEP is the logical entity or place on a server that makes admission control and policy decisions in response to a request from a user wanting to access a resource on a computer or network server.
PHP	Hypertext Preprocessor (PHP). PHP is an open source, server-side, HTML embedded scripting language used to create dynamic Web pages.
PIF	Protocol Interface Function. A PIF service encapsulates an interface with an external device or service
PIF Agent	An Agent that acts an adaptor between the system and an external device or service.
Pin	An input or output from an operator. The pin serves as a route endpoint and holds a single input or output value. For example, an operator that needs a username and password as input has two input pins; one for the username; the other, the password.
PMF	Profile Management Function. The Profile Management Function (PMF) activates and deactivates network profiles on subscriber sessions.
Point-to-Point Protocol Over ATM	PPPoA. PPPoA relies on two widely accepted standards: PPP and ATM. It is an end-to-end asymmetric digital subscriber line (ADSL) architecture.

Table 4-1. Terms and Definitions.

Term	Definition
Point-to-Point Termination Aggregation	PTA. This is a method of aggregating IP traffic by terminating PPP sessions and amassing the IP traffic into a single routing domain.
Policy	A flow comprising a rule or set of rules that take a specific action provided by an ISP for its subscribers. For example, a policy for subscriber access directs how the system identifies a subscriber via user id, access type, and log in location. A policy performs an operation, based on input and returns the results of its action as output.
Policy Database	The database of policy objects that services access to make policy decisions.
Policy Decision Point	PDP. The PDP is a component of policy-based management. When a user tries to access a file or other resource on a system using policy-based access management, the PDP decides whether or not to authorize the user based on user attributes.
Policy Enforcement Point	PEP. The PEP is the logical entity or place on a server that makes admission control and policy decisions in response to a request from a user wanting to access a resource on a computer or network server.
Policy Engine	PE. The software that stores and manages user profile information, subscriber access records, policy rules; also known as the policy database.
Policy Function	Policy rules encapsulated in a TCL agent *execute* function.
Policy Repository	The Policy Repository BPM stores all persistent data associated with customers and services. It utilizes industry-standard database technology that allows any of the underlying system elements to interrogate it.
Pool	A pool represents a range of IP addresses. A BRAS handles one or more address ranges. A Resource Controller potentially handles multiple BRASs. So a typical Resource Controller can handle multiple ranges of IP addresses (multiple pools).
PPPoA	Point-to-Point Protocol Over Asynchronous Transfer Mode. PPPoA relies on two widely accepted standards: PPP and ATM. It is an end-to-end asymmetric digital subscriber line (ADSL) architecture.
Presence Director	The Presence Director is an optional, modified, Director service that handles receives session requests and distributes them to the appropriate Resource Controllers.
Profile	A profile is a procedure for changing a set of related network elements for a given purpose, for example, increasing the bandwidth associated with a network session.
Profile Management Function	PMF. The Profile Management Function (PMF) activates and deactivates network profiles on subscriber sessions.
Property	The parameter or characteristic of an agent or device.
Protocol Interface Function	PIF. A PIF service encapsulates an interface with an external device or service.
PTA	Point-to-Point Termination Aggregation. This is a method of aggregating IP traffic by terminating PPP sessions and amassing the IP traffic into a single routing domain.

Table 4-1. Terms and Definitions.

Term	Definition
Q	
QoS	Quality of Service. QoS specifies a guaranteed throughput level that allows providers to guarantee to their customers that end-to-end latency will not exceed a specified level.
Quality of Service	QoS. QoS specifies a guaranteed throughput level that allows service providers to guarantee to their customers that end-to-end latency will not exceed a specified level.
R	
RACS	Resource and Admission Control Subsystem. RACS consists of the Policy Decision Function (PDF) and Access-RAC Function (A-RACF), which controls QoS within the access network.
RADIUS	Remote Authentication Dial-In User Service. RADIUS is a client/server protocol enabling remote access server communication with a central server to authenticate dial-in users and authorize their access to the requested system or service. RADIUS allows a company to maintain user profiles in a central database and set up a policy that can be applied at a single administered network point.
Realm	A realm represents a collection of information, stored in the database, that should be transferred, as a unit, between BPM systems. The realm defines a unit for intersystem communication and improves performance by restricting lookups and updates against smaller data sets.
Remote Authentication Dial-in User Service	RADIUS. RADIUS is a client/server protocol enabling remote access server communication with a central server to authenticate dial-in users and authorize their access to the requested system or service. RADIUS allows a company to maintain user profiles in a central database and set up a policy that can be applied at a single administered network point.
Remote Method Invocation	RMI. RMI is the basis of distributed object computing in the Java environment. It defines how Java components can interoperate in a Java environment.
Resource	A resource is any device or other item that can be used. Devices such as printers and disk drives are resources. Memory is also a resource. In many operating systems, a resource is specifically data or routines that are available to programs. These are also called system resources.
Resource and Admission Control Subsystem	RACS. RACS consists of the Policy Decision Function (PDF) and Access-RAC Function (A-RACF), which controls QoS within the access network.
Resource Controller	A Resource Controller is a stateful installation that tracks resource utilization for the system.
Resource Realm	A Resource Realm represents a BRAS device and its connected CPE equipment. The Resource Realm is provisioned on the Topology Database Server and distributed to the Resource Controller that coordinates activity for that BRAs. At runtime, the Resource Realm stores capacity and usage information required to perform CAC decisions.

Table 4-1. Terms and Definitions.

Term	Definition
RMI	Remote Method Invocation. RMI is the basis of distributed object computing in the Java environment. It defines how Java components can interoperate in a Java environment.
Role	A role is as a functional category for device types and device instances. For example, *bras* and *dpi* are roles.
Role-based Dependency	A dependency in which a service designer indicates that multiple service elements support the same interface. The designer defines different roles and assigns the required service interfaces to each. The different roles are added to the dependency list for the service and operators are clearly marked to indicate their assigned role.
Route	A route is a path between operators.
Rule	Criteria applied to the objects and methods of a business system to determine how objects and methods are used by, or for, a given system subscriber. A flow comprises a rule or set of rules. Rules prescribe terms and conditions for a specific action provided by an ISP for its subscribers. One rule can call another rule.
S	
S-VLAN	Stacked VLAN. An S-VLAN provides a two-level S-VLAN tag structure that extends the VLAN ID space to more than 16 million VLANs.
SAV	Service Admin view. In the BPS and BPDS graphical user interfaces to the BPM, this is the view where you can perform service tasks.
Schema	A set of rules and syntax for storing data.
SDV	Service Design view. In the BPS and BPDS graphical user interfaces to the BPM, this is the view where you can design services.
SE	Service Engine. SE is an unassigned and unconfigured system. It is also known as the backend.
Service	An application, created by the BPM designer, that maps business models and operational procedures directly into IP services, executable by their customers, for example, video on demand or automatic backup. A service comprises objects (agent, controller, function, switch, or other service) and can comprise one or more flows.
Service Admin View	SAV. In the BPS and BPDS graphical user interfaces to the BPM, this is the view where you can perform service tasks.
Service Configuration	The information needed to construct a service. The service configuration specifies agent configurations for each function in the service type. The BPM designer creates the service configuration.
Service Dependency	The dependencies of a service, created by the service designer. The designer builds a service by defining data-flows that use operators from multiple objects, including agents and other services. The designer builds a service upon a concrete set of agents and services. If a service is portable across different agents and services, the designer specifies any constraints on the concrete instances and specifies the interfaces that those concrete instances must support.
Service Design View	SDV. In the BPS and BPDS graphical user interfaces to the BPM, this is the view where you can design services.
Service Engine	SE. SE is the generic term for an unassigned and unconfigured system. it is also known as the backend.

Table 4-1. Terms and Definitions.

Term	Definition
Service Interface Dependency	If a service uses a particular service interface, but does not require that a specific object provide the service interface, the service designer can add the service interface as a dependency. Here, the service interface operators are available for use in the current service, but the object that provides the interface is determined later.
Service Instance	The running of a service type created by the subscriber.
Service Level Agreement	SLA. An SLA is a contract between an ASP and the end user that stipulates the required level of service and its fee.
Service Palette	The agent types available to a service.
Service Profile	A collection of services and information about service execution.
Service Provider	SP. This is the provider of Internet connectivity services.
Service Type	The definition of what agent types are required for a service; the defined flow of data between functions of agent types. The service designer creates the service type.
Servlet	An applet that runs on a server. Usually refers to a Java applet that runs within a Web server environment. Analogous to a Java applet that runs within a Web browser environment.
Session Management Application	SMA. Within the Session Manager, the SMA encapsulates customer-specific business logic for managing network sessions.
Session Management Function	SMF. The SMF encapsulates customer-specific business logic applied to network sessions. Abstracted from specific protocols and devices used in the network through the DAF and PIF layers, the SMF notifies applications of session state changes.
Session Manager	SM. The SM provides a framework for tracking user sessions connecting to the network.
Session Realm	A Session Realm stores Session Manager contexts and assists in the decision-making process during network adaptation.
Session Storage Function	SSF. The SSF provides access to the Session Information Model.
SF	Statistics Function. The SF records and queries system statistics and provides a location for various components to store runtime state statistics.
Shared Secret	An authentication string that ensures security between devices. KERBEROS is an instance of a shared-secret authentication protocol.
SIF	Signaling Interface Function (SIF): The SIF sends QoS requests from an application to the Director ACF. If more than one Director exists, an external Load Balancer selects a Director. The SIF receives replies from Director ACFs and forwards them to the application.
Signaling Interface Function	SIF. The SIF sends QoS requests an application to the Director ACF. If more than one Director exists, an external Load Balancer selects a Director. The SIF receives replies from Director ACFs and forwards them to the application.

Table 4-1. Terms and Definitions.

Term	Definition
Simple Object Access Protocol	SOAP. This is a lightweight XML-based messaging protocol that encodes the information in Web service request and response messages before sending them over a network. SOAP messages are independent of any operating system or protocol and may be transported using a variety of Internet protocols, including SMTP, MIME, and HTTP.
Simple Type	In the BPDS Object manager tool, a simple type is similar to data type, except it can express with enumerations.
Simple Network Management Protocol	SNMP. A protocol by which networked devices are periodically polled for information as part of a network management system.
SLA	Service Level Agreement. An SLA is a contract between an ASP and the end user that stipulates a required level of service and its fee.
SM	Session Manager. The SM provides a framework for tracking user sessions connecting to the network.
SMA	Session Management Application. Within the Session Manager, the SMA encapsulates customer-specific business logic for managing network sessions.
SMF	Session Management Function. The SMF encapsulates customer-specific business logic applied to network sessions. Abstracted from specific protocols and devices used in the network through the DAF and PIF layers, the SMF notifies applications of session state changes.
SNMP	Simple Network Management Protocol. A protocol by which networked devices are periodically polled for information as part of a network management system.
SOAP	Simple Object Access Protocol. This is a lightweight XML-based messaging protocol that encodes the information in Web service request and response messages before sending them over a network. SOAP messages are independent of any operating system or protocol and may be transported using a variety of Internet protocols, including SMTP, MIME, and HTTP.
SP	Service Provider. This is the provider of Internet connectivity services.
Specification	A type definition that includes interface definitions, configuration schemas, and binding information.
SQL	Structured Query Language. SQL is a standardized query language for requesting information from a database. SQL enables several users on a local-area network to access the same database simultaneously.
SSF	Session Storage Function. The SSF provides access to the Session Information Model.
Stacked VLAN	S-VLAN. An S-VLAN provides a two-level S-VLAN tag structure that extends the VLAN ID space to more than 16 million VLANs.
Standby BPM	In a pair of BPMs, the standby BPM constantly monitors the health of the active BPM to assess its ability to process requests. If the active BPM is not viable, the standby BPM becomes the active.
Statistics Function	SF. The SF records and queries system statistics and provides a location for various components to store runtime state statistics.
Status	A status is a condition used in conjunction with network sessions.

Table 4-1. Terms and Definitions.

Term	Definition
Structured Query Language	SQL. SQL is a standardized query language for requesting information from a database. SQL enables several users on a local-area network to access the same database simultaneously.
Subscriber	A customer of a service provider. The service provider delivers a variety of online services, including e-mail, stock quotes, news, and online forums.
Subscriber Profile	A table entry containing information, such as authentication, authorization, and location on a specific subscriber.
Super Operator	A reusable flow that other flows can call. To the other flows, the super operator appears as an operator that they can call and insert on any route.
Switch	A device that filters and forwards packets between LAN segments. Switches operate at the data link layer and the network layer of the OSI Reference Model.
Super User	The term denotes the highest level of user privilege. It allows unlimited access to a system. Usually, super user is the highest level of privilege for applications, as opposed to operating or network systems.
T	
TAF	Topology Awareness Function. The TAF extracts and reacts to changes in the underlying network. The information can be read from provisioning files or received from the TDS.
TISPAN	Telecommunications and Internet Services and Protocol for Advanced Networking. TISPAN is the ETSI core competence center for fixed networks and for migration from switched circuit networks to packet-based networks with an architecture that can serve in both. TISPAN is responsible for all aspects of standardization for present and future converged networks.
Telecommunications and Internet Services and Protocol for Advanced Networking	TISPAN. TISPAN is the ETSI core competence center for fixed networks and for migration from switched circuit networks to packet-based networks with an architecture that can serve in both. TISPAN is responsible for all aspects of standardization for present and future converged networks.
Topology Awareness Function	TAF. The TAF extracts and reacts to changes in the underlying network. The information can be read from provisioning files or received from the TDS.
Topology Database Server	In resilient pairs, Topology Database Servers maintain the global topology database for the system as a whole. The Director detects delayed response times or dropped requests and notifies the Topology Database Server. The Topology Database Server initiates Resource Controller failover when necessary.
Topology Store Function	TSF. The TSF maintains the TIM for a given BP Resource Controller system component.
Transaction Remote Procedure Call	TRPC. The TRPC protocol is the interface between Cisco BPM components.
TRPC	Transaction Remote Procedure Call.The TRPC protocol is the interface between Cisco BPM components.

Table 4-1. Terms and Definitions.

Term	Definition
TSF	Topology Store Function. The TSF maintains the TIM for a given BP Resource Controller system component.
Type	A BPM component group that has a unique specification. It may have an implementation, and it may have one or more configurations and instances.
U	
V	
VC	Virtual Circuit. A connection between two devices that acts as though it's a direct connection even though it may physically be circuitous.
Virtual Circuit	VC. A VC is a connection between two devices that acts as though it's a direct connection even though it may physically be circuitous.
Virtual LAN	VLAN. A network of computers that behave as if connected to the same wire even though they can be physically located on different segments of a LAN. VLANs are configured through software rather than hardware and extremely flexible.
Virtual Path	VP. A VC is a set of links across an ATM network between two specified end points.
Virtual Private Network	VPN. A VPN is constructed using public wires to connect nodes. A number of systems exist that enable the creation of networks using the Internet as the medium for transporting data. They use security mechanisms to ensure that only authorized users can access the network and data cannot be intercepted.
VLAN	Virtual LAN. A network of computers that behave as if they are connected to the same wire even though they may be physically located on different segments of a LAN. VLANs are configured through software rather than hardware and are extremely flexible.
Voice-over-IP	VoIP. Voice delivered using the Internet Protocol.
VoIP	Voice-over-IP. Voice delivered using the Internet Protocol.
VP	Virtual Path. VP. A VP is a set of link across an ATM network between two specified end points.
VPN	Virtual Private Network. A VPN is constructed using public wires to connect nodes. A number of systems exist that enable the creation of networks using the Internet as the medium for transporting data. They use security mechanisms to ensure that only authorized users can access the network and data cannot be intercepted.
W	
WDSL	Wireless Digital Subscriber Line. WDSL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. It is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate.

Table 4-1. Terms and Definitions.

Term	Definition
Wireless Digital Subscriber Line	WDSL. WDSL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. It is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate.
Workspace	The BPDS area where the designer visually programs services.

Appendix B - SCE Error Codes

Overview

Table 4 contains the possible SCE error codes and their descriptions. Review these error codes to interpret the returned *OperationException*. Extract the error code using the *getErrorCode* method. The error code enumeration is in the *com.scms.api.sce.SCESubscriberApi* interface.

Table 4. Error Codes and Descriptions

Error Code	Description
ERROR_CODE_SUBSCRIBER_DOES_NOT_EXIST	The subscriber on which the operation is performed does not exist in the SCE database.
ERROR_CODE_INVALID_PARAMETER	One of the parameters of the operation was invalid (networked, policy property etc.)
ERROR_CODE_NO_APPLICATION_INSTALLED	Application is required to perform the operation
ERROR_CODE_FATAL_EXCEPTION	Too many exceptions occurred at the SCE during the operation
ERROR_CODE_RESOURCE_SHORTAGE	Internal error.
ERROR_CODE_OPERATION_ABORTED	Internal error.
ERROR_CODE_ARRAY_ACCESS	Internal error.
ERROR_CODE_ATTRIBUTE_NOT_FOUND	Internal error.
ERROR_CODE_CLASS_CAST	Internal error.
ERROR_CODE_CLASS_NOT_FOUND	Internal error.
ERROR_CODE_CLIENT_INTERNAL_ERROR	Internal error.
ERROR_CODE_CLIENT_OUT_OF_THREADS	Internal error.
ERROR_CODE_ILLEGAL_STATE	Internal error.
ERROR_CODE_OBJECT_NOT_FOUND	Internal error.
ERROR_CODE_OPERATION_NOT_FOUND	Internal error.
ERROR_CODE_OUT_OF_MEMORY	Internal error.
ERROR_CODE_RUNTIME	Internal error.
ERROR_CODE_NULL_POINTER	Internal error.
ERROR_CODE_UNKNOWN	Internal error.

Appendix C - API Classes

Overview

This appendix contains the classes supplied user in the SCE Subscriber API package.

Package com.scms.api.sce.prpc

- *PRPC_SCESubscriberApi* (class) - main API class.

Indications Listeners

- *LoginPullListener* (interface)
- *LogoutListener* (interface)
- *QuotaListener* (interface)

Connection Monitoring

- *ConnectionListener* (interface)

Operations Result Handling

- *OperationException* (class)
- *SCESubscriberApi* (interface)
- *OperationArguments* (class)
- *OperationResultHandler* (interface)

Package com.scms.common

The package *com.scms.common* contains all data types used by the API.

- *Login_BULK* (class)
- *LoginPullResponse_BULK* (class)
- *NetworkAndSubscriberID_BULK* (class)
- *PolicyProfile_BULK* (class)
- *SubscriberID_BULK* (class)
- *SubscriberData* (class)
- *SCAS_BB_Quota* (class)
- *SCAS_BB_QuotaOperation* (class)
- *NetworkID* (class)
- *PolicyProfile* (class)

Index