SRC-PE Software

Solutions Guide

Release 1.0.x

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, CA 94089
USA
408-745-2000
www.juniper.net

Part Number: 530-020347-01, Revision A00

SRC-PE Software Solutions Guide, Release 1.0.x
Writing: Linda Creed, Justine Kangas, Betty Lew
Editing: Fran Mues
Illustration: Nathaniel Woodward
Cover Design: Edmonds Design

Revision History
6 April 2007—Revision 1

Software License

The terms and conditions for using this software are described in the software license contained in the acknowledgment to your purchase order or, to the extent applicable, to any reseller agreement or end-user purchase agreement executed between you and Juniper Networks. By using this software, you indicate that you understand and agree to be bound by those terms and conditions.

Generally speaking, the software license restricts the manner in which you are permitted to use the software and may contain prohibitions against certain uses. The software license may state conditions under which the license is automatically terminated. You should consult the license for further details.

For complete product documentation, please see the Juniper Networks Web site at www.juniper.net/techpubs.

END USER LICENSE AGREEMENT

READ THIS END USER LICENSE AGREEMENT ("AGREEMENT") BEFORE DOWNLOADING, INSTALLING, OR USING THE SOFTWARE. BY DOWNLOADING, INSTALLING, OR USING THE SOFTWARE OR OTHERWISE EXPRESSING YOUR AGREEMENT TO THE TERMS CONTAINED HEREIN, YOU (AS CUSTOMER OR IF YOU ARE NOT THE CUSTOMER, AS A REPRESENTATIVE/AGENT AUTHORIZED TO BIND THE CUSTOMER) CONSENT TO BE BOUND BY THIS AGREEMENT. IF YOU DO NOT OR CANNOT AGREE TO THE TERMS CONTAINED HEREIN, THEN (A) DO NOT DOWNLOAD, INSTALL, OR USE THE SOFTWARE, AND (B) YOU MAY CONTACT JUNIPER NETWORKS REGARDING LICENSE TERMS.

1. The Parties. The parties to this Agreement are Juniper Networks, Inc. and its subsidiaries (collectively "Juniper"), and the person or organization that originally purchased from Juniper or an authorized Juniper reseller the applicable license(s) for use of the Software ("Customer") (collectively, the "Parties").

2. The Software. In this Agreement, "Software" means the program modules and features of the Juniper or Juniper-supplied software, and updates and releases of such software, for which Customer has paid the applicable license or support fees to Juniper or an authorized Juniper reseller. "Embedded Software" means Software which Juniper has embedded in the Juniper equipment.

3. License Grant. Subject to payment of the applicable fees and the limitations and restrictions set forth herein, Juniper grants to Customer a non-exclusive and non-transferable license, without right to sublicense, to use the Software, in executable form only, subject to the following use restrictions:

a. Customer shall use the Embedded Software solely as embedded in, and for execution on, Juniper equipment originally purchased by Customer from Juniper or an authorized Juniper reseller.

b. Customer shall use the Software on a single hardware chassis having a single processing unit, or as many chassis or processing units for which Customer has paid the applicable license fees; provided, however, with respect to the Steel-Belted Radius or Odyssey Access Client software only, Customer shall use such Software on a single computer containing a single physical random access memory space and containing any number of processors. Use of the Steel-Belted Radius software on multiple computers requires multiple licenses, regardless of whether such computers are physically contained on a single chassis.

c. Product purchase documents, paper or electronic user documentation, and/or the particular licenses purchased by Customer may specify limits to Customer's use of the Software. Such limits may restrict use to a maximum number of seats, registered endpoints, concurrent users, sessions, calls, connections, subscribers, clusters, nodes, realms, devices, links, ports or transactions, or require the purchase of separate licenses to use particular features, functionalities, services, applications, operations, or capabilities, or provide throughput, performance, configuration, bandwidth, interface,

processing, temporal, or geographical limits. In addition, such limits may restrict the use of the Software to managing certain kinds of networks or require the Software to be used only in conjunction with other specific Software. Customer's use of the Software shall be subject to all such limitations and purchase of all applicable licenses.

d. For any trial copy of the Software, Customer's right to use the Software expires 30 days after download, installation or use of the Software. Customer may operate the Software after the 30-day trial period only if Customer pays for a license to do so. Customer may not extend or create an additional trial period by re-installing the Software after the 30-day trial period.

e. The Global Enterprise Edition of the Steel-Belted Radius software may be used by Customer only to manage access to Customer's enterprise network. Specifically, service provider customers are expressly prohibited from using the Global Enterprise Edition of the Steel-Belted Radius software to support any commercial network access services.

The foregoing license is not transferable or assignable by Customer. No license is granted herein to any user who did not originally purchase the applicable license(s) for the Software from Juniper or an authorized Juniper reseller.

4. Use Prohibitions. Notwithstanding the foregoing, the license provided herein does not permit the Customer to, and Customer agrees not to and shall not: (a) modify, unbundle, reverse engineer, or create derivative works based on the Software; (b) make unauthorized copies of the Software (except as necessary for backup purposes); (c) rent, sell, transfer, or grant any rights in and to any copy of the Software, in any form, to any third party; (d) remove any proprietary notices, labels, or marks on or in any copy of the Software or any product in which the Software is embedded; (e) distribute any copy of the Software to any third party, including as may be embedded in Juniper equipment sold in the secondhand market; (f) use any 'locked' or key-restricted feature, function, service, application, operation, or capability without first purchasing the applicable license(s) and obtaining a valid key from Juniper, even if such feature, function, service, application, operation, or capability is enabled without a key; (g) distribute any key for the Software provided by Juniper to any third party; (h) use the Software in any manner that extends or is broader than the uses purchased by Customer from Juniper or an authorized Juniper reseller; (i) use the Embedded Software on non-Juniper equipment; (j) use the Software (or make it available for use) on Juniper equipment that the Customer did not originally purchase from Juniper or an authorized Juniper reseller; (k) disclose the results of testing or benchmarking of the Software to any third party without the prior written consent of Juniper; or (l) use the Software in any manner other than as expressly provided herein.

5. Audit. Customer shall maintain accurate records as necessary to verify compliance with this Agreement. Upon request by Juniper, Customer shall furnish such records to Juniper and certify its compliance with this Agreement.

6. Confidentiality. The Parties agree that aspects of the Software and associated documentation are the confidential property of Juniper. As such, Customer shall exercise all reasonable commercial efforts to maintain the Software and associated documentation in confidence, which at a minimum includes restricting access to the Software to Customer employees and contractors having a need to use the Software for Customer's internal business purposes.

7. Ownership. Juniper and Juniper's licensors, respectively, retain ownership of all right, title, and interest (including copyright) in and to the Software, associated documentation, and all copies of the Software. Nothing in this Agreement constitutes a transfer or conveyance of any right, title, or interest in the Software or associated documentation, or a sale of the Software, associated documentation, or copies of the Software.

8. Warranty, Limitation of Liability, Disclaimer of Warranty. The warranty applicable to the Software shall be as set forth in the warranty statement that accompanies the Software (the "Warranty Statement"). Nothing in this Agreement shall give rise to any obligation to support the Software. Support services may be purchased separately. Any such support shall be governed by a separate, written support services agreement. TO THE MAXIMUM EXTENT PERMITTED BY LAW, JUNIPER SHALL NOT BE LIABLE FOR ANY LOST PROFITS, LOSS OF DATA, OR COSTS OR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, OR FOR ANY SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THIS AGREEMENT, THE SOFTWARE, OR ANY JUNIPER OR JUNIPER-SUPPLIED SOFTWARE. IN NO EVENT SHALL JUNIPER BE LIABLE FOR DAMAGES ARISING FROM UNAUTHORIZED OR IMPROPER USE OF ANY JUNIPER OR JUNIPER-SUPPLIED SOFTWARE. EXCEPT AS EXPRESSLY PROVIDED IN THE WARRANTY STATEMENT TO THE EXTENT PERMITTED BY LAW, JUNIPER DISCLAIMS ANY AND ALL WARRANTIES IN AND TO THE SOFTWARE (WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE), INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT DOES JUNIPER WARRANT THAT THE SOFTWARE, OR ANY EQUIPMENT OR NETWORK RUNNING THE SOFTWARE, WILL OPERATE WITHOUT ERROR OR INTERRUPTION, OR WILL BE FREE OF VULNERABILITY TO INTRUSION OR ATTACK. In no event shall Juniper's or its suppliers' or licensors' liability to Customer, whether in contract, tort (including negligence), breach of warranty, or otherwise, exceed the price paid by Customer for the Software that gave rise to the claim, or if the Software is embedded in another Juniper product, the price paid by Customer for such other product. Customer acknowledges and agrees that Juniper has set its prices and entered into this Agreement in reliance upon the disclaimers of warranty and the limitations of liability set forth herein, that the same reflect an allocation of risk between the Parties (including the risk that a contract remedy may fail of its essential purpose and cause consequential loss), and that the same form an essential basis of the bargain between the Parties.

9. Termination. Any breach of this Agreement or failure by Customer to pay any applicable fees due shall result in automatic termination of the license granted herein. Upon such termination, Customer shall destroy or return to Juniper all copies of the Software and related documentation in Customer's possession or control.

10. Taxes. All license fees for the Software are exclusive of taxes, withholdings, duties, or levies (collectively "Taxes"). Customer shall be responsible for paying Taxes arising from the purchase of the license, or importation or use of the Software.

11. Export. Customer agrees to comply with all applicable export laws and restrictions and regulations of any United States and any applicable foreign agency or authority, and not to export or re-export the Software or any direct product thereof in violation of any such restrictions, laws or regulations, or without all necessary approvals. Customer shall be liable for any such violations. The version of the Software supplied to Customer may contain encryption or other capabilities restricting Customer's ability to export the Software without an export license.

12. Commercial Computer Software. The Software is "commercial computer software" and is provided with restricted rights. Use, duplication, or disclosure by the United States government is subject to restrictions set forth in this Agreement and as provided in DFARS 227.7201 through 227.7202-4, FAR 12.212, FAR 27.405(b)(2), FAR 52.227-19, or FAR 52.227-14(ALT III) as applicable.

13. Interface Information. To the extent required by applicable law, and at Customer's written request, Juniper shall provide Customer with the interface information needed to achieve interoperability between the Software and another independently created program, on payment of applicable fee, if any. Customer shall observe strict obligations of confidentiality with respect to such information and shall use such information in compliance with any applicable terms and conditions upon which Juniper makes such information available.

14. Third Party Software. Any licensor of Juniper whose software is embedded in the Software and any supplier of Juniper whose products or technology are embedded in (or services are accessed by) the Software shall be a third party beneficiary with respect to this Agreement, and such licensor or vendor shall have the right to enforce this Agreement in its own name as if it were Juniper. In addition, certain third party software may be provided with the Software and is subject to the accompanying license(s), if any, of its respective owner(s). To the extent portions of the Software are distributed under and subject to open source licenses obligating Juniper to make the source code for such portions publicly available (such as the GNU General Public License ("GPL") or the GNU Library General Public License ("LGPL")), Juniper will make such source code portions (including Juniper modifications, as appropriate) available upon request for a period of up to three years from the date of distribution. Such request can be made in writing to Juniper Networks, Inc., 1194 N. Mathilda Ave., Sunnyvale, CA 94089, ATTN: General Counsel. You may obtain a copy of the GPL at http://www.gnu.org/licenses/gpl.html, and a copy of the LGPL at http://www.gnu.org/licenses/lgpl.html.

15. Miscellaneous. This Agreement shall be governed by the laws of the State of California without reference to its conflicts of laws principles. The provisions of the U.N. Convention for the International Sale of Goods shall not apply to this Agreement. For any disputes arising under this Agreement, the Parties hereby consent to the personal and exclusive jurisdiction of, and venue in, the state and federal courts within Santa Clara County, California. This Agreement constitutes the entire and sole agreement between Juniper and the Customer with respect to the Software, and supersedes all prior and contemporaneous agreements relating to the Software, whether oral or written (including any inconsistent terms contained in a purchase order), except that the terms of a separate written agreement executed by an authorized Juniper representative and Customer shall govern to the extent such terms are inconsistent or conflict with terms contained herein. No modification to this Agreement nor any waiver of any rights hereunder shall be effective unless expressly assented to in writing by the party to be charged. If any portion of this Agreement is held invalid, the Parties agree that such invalidity shall not affect the validity of the remainder of this Agreement. This Agreement and associated documentation has been written in the English language, and the Parties agree that the English version will govern. (For Canada: Les parties aux présentés confirment leur volonté que cette convention de même que tous les documents y compris tout avis qui s'y rattaché, soient redigés en langue anglaise. (Translation: The parties confirm that this Agreement and all related documentation is and will be in the English language)).

Table of Contents

About This Guide xi

Objectives ... xi
Audience ... xi
Documentation Conventions ... xii
Related Juniper Networks Documentation ... xiii
Obtaining Documentation ... xv
Documentation Feedback ... xv
Requesting Support ... xv

Chapter 1 Managing Tiered and Premium Services with QoS on JUNOSe Routers 1

Overview of QoS on JUNOSe Routers ... 1
Dynamically Managing QoS Profiles ... 2
How QoS Profile Tracking Works ... 2
Identifying QoS Services ... 2
Determining the QoS Profile ... 3
Setting Up Policy Groups ... 4
Setting Up Services ... 5
Reestablishing Default QoS Profile ... 5
Example: How QTP Activates a QoS Service ... 5
Configuring QoS Profile-Tracking Plug-Ins with the SRC CLI ... 7
Configuring Search Filters for QoS Profile-Tracking Plug-Ins ... 9
Configuring QoS Profile-Tracking Plug-Ins with SDX Configuration Editor ... 10
Updating QoS Profile Data in the Directory ... 13
Using SDX Admin to Update QoS Profile Data ... 14
Using qosProfilePublish to Update QoS Profile Data ... 14
Searching for QoS Policy Data in the Directory ... 16
Using Policy Editor to Search for QoS Policy Information ... 17
Running Queries from Policy Editor ... 17
Examples ... 19
Using Policy Web Admin to Search for QoS Policy Information ... 20
Launching Policy Web Admin ... 21
Connecting to a Directory ... 22
Querying the Directory for QoS Information ... 23

Chapter 2 Managing Subscribers for a Wireless Roaming Environment 25

Overview of a Wireless Roaming Environment ... 25
Subscriber Access in a Wireless Roaming Environment ... 26
Configuring Subscriber Access for a Wireless Location ... 27
Configuring RADIUS Authentication ... 27
Configuring a Custom RADIUS Authentication Plug-In ... 27
Configuring the Flexible RADIUS Authentication Plug-In ... 27
Creating Subscriber Access to an ISP ... 30

Creating Web Access 31
Verifying Idle Timeout Properties for the SAE 32

Chapter 3 Configuring VoIP Services in an SRC Network 33

Overview of Session Management for VoIP Services 33
Accounting and Tracking 34
VoIP Call Setup 34
Configuring Policies and Services for VoIP 34
Activating VoIP Services for Assigned IP Subscribers 35
Setting Timeouts for Assigned IP Subscriber Sessions 36

Chapter 4 Providing Premium Services in a PCMM Environment 37

Overview of a PCMM Environment 37
References 38
PCMM Architecture 38
DOCSIS Protocol 39
Service Flows 39
Client Types 40
SRC Software in the PCMM Environment 42
Traffic Profiles 43
End-to-End QoS Architecture 43
Extending QoS to the Subscriber Edge Domain 44
Extending QoS to the Service Edge Domain 44
Provisioning End-to-End Services 45
Example for Videoconferencing Services 45
Example for Video-on-Demand Services 46
Using the SAE in a PCMM Environment 47
Logging In Subscribers and Creating Sessions 48
Assigned IP Subscribers 48
Event Notification from an IP Address Manager 49
SAE Communities 51
Storing Session Data 52
PCMM Record-Keeping Server Plug-In 52

Chapter 5 Configuring the SAE for a PCMM Environment with the SRC CLI 53

Overview of Configuring the SAE for a Cable Network Environment 53
Configuring the SAE to Manage PCMM Devices 54
Related Information 56
Setting Up SAE Communities 57
Configuring the SAE Community Manager 57
Related Information 58
Configuring SAE Properties for the Event Notification API 58
Related Information 59
Configuring Record-Keeping Server Peers for Plug-Ins 59
Related Information 60
Configuring PCMM Record-Keeping Server Plug-Ins 60
Related Information 62
Configuring CMTS-Specific RKS Plug-Ins 63
Related Information 63

Chapter 6 Configuring the SAE for a PCMM Environment with SDX Configuration Editor 65

Overview of Configuring the SAE for a Cable Network Environment....65
Configuring the SAE to Manage PCMM Devices with SDX Configuration Editor....66
Setting Up SAE Communities....69
Configuring the SAE Community Manager....69
Configuring SAE Properties for the Event Notification API....70
Configuring PCMM Record-Keeping Server Plug-Ins....71
Configuring RKS Peers....74
Configuring CMTS-Specific RKS Plug-Ins....75

Chapter 7 Adding Objects for CMTS Devices with the SRC CLI 77

Adding Objects for CMTS Devices with the SRC CLI....77
Creating Virtual Routers for the CMTS Device with the SRC CLI....78

Chapter 8 Adding Objects to the Directory with SDX Admin 81

Adding Objects for CMTS Devices to the Directory with SDX Admin....81
Creating a Virtual Router for the CMTS Device with SDX Admin....83
Configuration Parameters for Virtual Routers....84
Configuring SAE Communities....87

Chapter 9 Using the NIC Resolver in a PCMM Environment 89

Overview of Using the NIC Resolver in a PCMM Environment....89
Accessing the OnePopDynamicIp Configuration with the SRC CLI....89
Accessing the OnePopDynamicIp Configuration on a Solaris Platform....90

Chapter 10 Using IPSec to Protect Communications Between the SAE and CMTS Device 91

Overview of IPSec....91
Security Keys....92
Key Types....92
Key Management....93
IPSec Configuration for the SAE....93
Before You Configure IPSec....94
Protecting IPSec Configuration Properties....95
Configuring IPSec for the SAE....95
Configuring IPSec with SDX Configuration Editor....95
Configuring Host Properties....96
Configuring Connection Properties....97
Configuring IPSec Properties to Establish Key Exchange and SAs....99
Applying the IPSec Configuration....102
Changing IPSec Configuration....102
Configuring IPSec on a Remote System....102
Testing the IPSec Connection....103

Chapter 11 Using PCMM Policy Servers 105

Overview of the JPS....105
JPS Framework....106

JPS Interfaces 107
Application Manager to Policy Server Interface 107
Policy Server to RKS Interface 107
Policy Server to CMTS Interface 107
Before You Configure the JPS 107

Chapter 12 Configuring the JPS with the SRC CLI 109

Configuration Statements for the JPS 109
Configuring the JPS 111
Modifying the JPS Configuration 112
Configuring General Properties for the JPS 112
Specifying Policy Server Identifiers in Messages 113
Configuring Logging Destinations for the JPS 114
Configuring Logging Destinations to Store Messages in a File 114
Configuring Logging Destinations to Send Messages to System Logging Facility 115
Specifying Connections to the Application Managers 115
Specifying Connections to RKSs 116
Configuring RKS Pairs 119
Configuring RKS Pairs for Associated Application Managers 120
Specifying Connections to CMTS Devices 121
Modifying the Subscriber Configuration 124
Configuring Subscriber IP Pools as IP Address Ranges 124
Configuring Subscriber IP Pools as IP Subnets 125
Configuring the SAE to Interact with the JPS 125
Specifying Application Managers for the Policy Server 126
Specifying Application Manager Identifiers for Policy Servers 127
Adding Objects for Policy Servers to the Directory 128
Configuring Initialization Scripts 128
Enabling State Synchronization 129
Using the NIC Resolver 130
Managing the JPS 131
Starting the JPS 131
Restarting the JPS 131
Stopping the JPS 131
Displaying JPS Status 131

Chapter 13 Configuring the JPS on a Solaris Platform 133

Installing the JPS 133
Configuring the JPS for Time Change Event Notification 134
Modifying the Local Clock 134
Starting and Managing the JPS 135
Starting the JPS 135
Restarting the JPS 136
Stopping the JPS 136
Displaying JPS Status 136
Configuring the JPS 137
Configuring the SAE to Interact with the JPS on Solaris Platforms 137
Specifying Application Managers for the Policy Server 138
Specifying Application Manager Identifiers for Policy Servers 141
Adding Objects for Policy Servers to the Directory 142
Configuring Initialization Scripts 143
Enabling State Synchronization 144

Monitoring the JPS 145

Chapter 14 Monitoring the JPS with the SRC CLI 147

Monitoring the JPS 147
Viewing Server Process Information 147
Viewing JPS State 148
Viewing Performance Statistics for the JPS Interfaces 148
Viewing Network Connections for the Application Manager 148
Viewing Network Connections for the CMTS Device 148
Viewing Performance Statistics for the CMTS Locator 149
Viewing Message Handler Information 149

Chapter 15 Monitoring the JPS with the C-Web Interface 151

Viewing Information About JPS Server Process with the C-Web Interface 151
Viewing JPS AM Statistics with the C-Web Interface 152
Viewing JPS AM Connections with the C-Web Interface 153
Viewing JPS CMTS Statistics with the C-Web Interface 153
Viewing JPS CMTS Connections with the C-Web Interface 154
Viewing JPS CMTS Locator Statistics with the C-Web Interface 155
Viewing JPS Message Handler Statistics with the C-Web Interface 155
Viewing JPS Message Flow Statistics with the C-Web Interface 156
Viewing JPS RKS Statistics with the C-Web Interface 157

Chapter 16 Providing Packet Mirroring in the SRC Network 159

Overview of Packet Mirroring 159
Configuring Packet Mirroring 160
Creating the Script Service for Packet Mirroring 160
Configuring the Script Service for Packet Mirroring 162
Configuring Subscriptions to the Packet-Mirroring Service 164
Specifying Maximum Number of Peers 164
Example: Using the Sample Packet-Mirroring Application 164
Example: Packet Mirroring for PPP Subscribers 165
Example: Packet Mirroring for DHCP Subscribers 165
Configuring DHCP Subscriber Sessions 165
Disabling RADIUS Authentication for DHCP Subscribers 166
Defining RADIUS Attributes for Dynamic Authorization Requests with the API 166

Chapter 17 Configuring IPTV Services in an SRC Network 167

Overview of IPTV Service Applications 167
Installing the Sample IPTV Application 168
Configuring the Sample IPTV Application 168
Setting Up the IPTV Network 170
Configuring the SAE for the IPTV Application 171
Managing the Routers in an IPTV Network 171
Configuring IPTV Subscribers and Services 171
Configuring SRC-ACP as an External Plug-In for the IPTV Application 172
Configuring Event Publishers for the IPTV Application 173
Configuring the NIC as an External Plug-In for the IPTV Application 173

Configuring SRC-ACP for the IPTV Application....173
Defining SRC-ACP Properties for the IPTV Application....174
Defining the Sample Congestion Points for the IPTV Application....175
Configuring the NIC for the IPTV Application....176
Running the Sample IPTV Application....176

Chapter 18 Providing Services in IMS Networks 179

Overview of an IMS Environment....179
IMS and ETSI References....180
Abbreviations....181
IMS Layers....181
Signaling Protocol....182
ETSI-TISPAN Architecture....183
RACS Layer....183
Rq Interface....183
SPDF....184
A-RACF....184
SRC Software in the ETSI-TISPAN Architecture....185
SRC Software in the IMS Environment....186
Installing and Configuring the IMS Software....187
Configuration Fields for DIAMETER Peers....187
Configuring Logging Destinations....188
Bootstrap Properties for IMS....190
Starting the IMS Process to Provide the A-RACF Rq Interface....191
Stopping the IMS Process to Provide the A-RACF Rq Interface....192
Cleaning the IMS Log Files....192
Testing and Demonstrating the A-RACF Rq Interface....192
Rq Interface Messaging....193
Configuring Policies for IMS....193
Enabling Expansion of JUNOSe Classify-Traffic Conditions....194
Enable JUNOSe Classifier Expansion Field....195

Chapter 19 Providing Prepaid Services 197

Overview of Prepaid Services Demo....197
Account Server....198
Time-Based Services....198
Volume-Based Services....198
Installing and Configuring the Prepaid Services Demo....199
Installing the Account Server....199
Configuring the Account Server....200
Publishing the Object References....200
Manual Configuration....201
Starting the Account Server....201
Stopping the Account Server....202
Configuring the SAE for the Prepaid Plug-In....202
Configuring the Prepaid Services....203
Deploying the Prepaid Account Administration Application....203
Configuring the Prepaid Account Administration Application....203
Managing Prepaid Accounts....203
Accessing the Prepaid Account Administration Application....203
Administering Accounts....204

Index 205

About This Guide

This preface provides the following guidelines for using the *SRC-PE Software Solutions Guide*.

- Objectives on page xi
- Audience on page xi
- Documentation Conventions on page xii
- Related Juniper Networks Documentation on page xiii
- Obtaining Documentation on page xv
- Documentation Feedback on page xv
- Requesting Support on page xvi

Objectives

This guide provides information about how to configure the Session and Resource Control (SRC) software in a number of specific use scenarios.

NOTE: If the information in the latest *SRC Release Notes* differs from the information in this guide, follow the *SRC Release Notes*.

Audience

This guide is intended for experienced system and network specialists working with JUNOSe routers and JUNOS routing platforms in an Internet access environment. We assume that readers know how to use the routing platforms, directories, and RADIUS servers that they will deploy in their SRC networks. For users who deploy the SRC software on a Solaris platform, we also assume that readers are familiar with the Lightweight Directory Access Protocol (LDAP) and the UNIX operating system.

If you are using the SRC software in a cable network environment, we assume that you are familiar with the *PacketCable Multimedia Specification* (PCMM) as defined by Cable Television Laboratories, Inc. (CableLabs) and with the Data-over-Cable Service Interface Specifications (DOCSIS) 1.1 protocol. We also assume that you are familiar with operating a multiple service operator (MSO) multimedia-managed IP network.

Documentation Conventions

The sample screens used throughout this guide are representations of the screens that are displayed when you install and configure the SRC software. The actual screens may differ.

For convenience and clarity, the installation and configuration examples show default file paths. If you do not accept the installation defaults, your paths will vary from the examples.

Table 1 defines notice icons used in this guide. Table 2 defines text conventions used throughout the documentation.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury.

Table 2: Text Conventions

Convention	Description	Examples
Bold typeface	■ Represents keywords, scripts, and tools in text. ■ Represents a GUI element that the user selects, clicks, checks, or clears.	■ Specify the keyword **exp-msg**. ■ Run the **install.sh** script. ■ Use the **pkgadd** tool. ■ To cancel the configuration, click **Cancel**.
Bold sans serif typeface	Represents text that the user must type.	user@host# **set cache-entry-age** *cache-entry-age*

`Monospace sans serif typeface`	Represents information as displayed on your terminal's screen, such as CLI commands in output displays.	`nic-locators {`
		`  login {`
		`  resolution {`
		`  resolver-name /realms/login/A1;`
		`  key-type LoginName;`
		`  value-type SaeId;`
		`  }`

Regular sans serif typeface	■ Represents configuration statements. ■ Indicates SRC CLI commands and options in text. ■ Represents examples in procedures. ■ Represents URLs.	■ system ldap server { stand-alone; ■ Use the request sae modify device failover command with the force option. ■ user@host# . . . ■ http://www.juniper.net/techpubs/software/management/sdx/api-index.html
Italic sans serif typeface	Represents variables in SRC CLI commands.	user@host# **set local-address** *local-address*
Angle brackets	In text descriptions, indicate optional keywords or variables.	Another runtime variable is < gfwif > .

Table 2: Text Conventions (continued)

Convention	Description	Examples
Key name	Indicates the name of a key on the keyboard.	Press Enter.
Key names linked with a plus sign (+) .	Indicates that you must press two or more keys simultaneously.	Press Ctrl + b.
Italic typeface	■ Emphasizes words. ■ Identifies chapter, appendix, and book names. ■ Identifies distinguished names. ■ Identifies files, directories, and paths in text but not in command examples.	■ There are two levels of access: *user* and *privileged.* ■ *Chapter 2, Services.* ■ *o = Users, o = UMC* ■ The */etc/default.properties* file.
Backslash	At the end of a line, indicates that the text wraps to the next line.	Plugin.radiusAcct-1.class = \ net.juniper.smgt.sae.plugin\ RadiusTrackingPluginEvent
Words separated by the | symbol	Represent a choice to select one keyword or variable to the left or right of this symbol. (The keyword or variable may be either optional or required.)	diagnostic | line

Related Juniper Networks Documentation

With each SRC software release, we provide the *SRC Documentation CD,* which contains the documentation described in Table 3.

With each SRC Application Library release, we provide the *SRC Application Library CD*. This CD contains both the software applications and the *SRC Application Library Guide*.

The C-Web interface, which is based on the J-Web interface, is available for monitoring C-series platforms and the SRC software. For general information about the J-Web interface, see the *J-Web Interface User Guide*.

A complete list of abbreviations used in this document set, along with their spelled-out terms, is provided in the *SRC Getting Started Guide*.

Table 3: Juniper Networks C-series and SRC Technical Publications

Document	Description
Core Documentation Set	
C-series Hardware Guide	Describes the hardware platforms and how to install, maintain, replace, and troubleshoot them. The guide also includes specifications.
SRC-PE Getting Started Guide	Describes the SRC software and explains how to set up an initial configuration and manage a C-series platform. The guide describes how to set up and start the SRC CLI and C-Web, as well as other SRC configurations. It provides information about setting up an initial SRC configuration on a Solaris platform. The guide also describes how to upgrade the SRC software and how to use the SRC configuration tools. It includes reference material for the SRC documentation.

Table 3: Juniper Networks C-series and SRC Technical Publications (continued)

Document	Description
SRC-PE CLI User Guide	Describes how to use the SRC CLI, configure and monitor the platform with the CLI, and control the CLI environment. The guide also describes how to manage SRC components with the CLI.
SRC-PE Network Guide: SAE, Juniper Networks Routers, and NIC	Describes how to use and configure the SAE and the NIC. This guide also provides detailed information for using JUNOSe routers and JUNOS routing platforms in the SRC network.
SRC-PE Integration Guide: Network Devices, Directories, and RADIUS Servers	Describes how to integrate external components—network devices, directories, and RADIUS servers—into the SRC network. The guide provides detailed information about integrating specific models of the external components.
SRC-PE Services and Policies Guide	Describes how to work with services and policies. The guide provides an overview, configuration procedures, and management information. The guide also provides information about the SRC tools for configuring policies.
SRC-PE Subscribers and Subscriptions Guide	Describes how to work with residential and enterprise subscribers and subscriptions. The guide provides an overview, configuration procedures, and management information. This guide also provides information about the sample residential portals and enterprise service portals, including the Enterprise Manager Portal.
SRC-PE Monitoring and Troubleshooting Guide	Describes how to use logging, the SNMP agent, the SRC CLI, and the C-Web interface to monitor and troubleshoot SRC components. This guide also describes the SNMP traps.
SRC-PE Solutions Guide	Provides high-level instructions for SRC implementations. The guide documents the following scenarios: managing QoS services on JUNOSe routers; managing subscribers in a wireless roaming environment; providing voice over IP (VoIP) services; integrating the SRC software in a PCMM environment, including the use of the Juniper Policy Server (JPS); mirroring subscriber traffic on JUNOSe routers; demonstrating network resource management features in a sample IP television (IPTV) application; and demonstrating the integration of prepaid services in a sample application.
SRC-PE CLI Command Reference, Volume 1 *SRC-PE CLI Command Reference, Volume 2*	Together constitute information about command and statement syntax; descriptions of commands, configuration statements, and options; editing level of statement options; and a history of when a command was added to the documentation.
SRC-PE Comprehensive Index	Provides a complete index of the SRC guides, excluding the *C-series Hardware Guide* and the *SRC CLI Command Reference*.
J-Web User Interface Guide	Provides general information about the J-Web interface.
Application Library	
SRC Application Library Guide	Describes how to install and work with applications that you can use to extend the capabilities of the SRC software. The guide documents the following applications: SRC-SG (SOAP Gateway) Web applications, applications to integrate the Juniper Networks Intrusion Detection and Protection (IDP) software into an SRC-managed environment, an application to provide endpoint security by integrating Juniper Networks Instant Virtual Extranet (IVE) Host Checker, a traffic-mirroring Web application, an application to integrate IP address managers with the SAE, an application to provide tracking and QoS control at the application level by integrating the SRC software with the Ellacoya deep packet inspection (DPI) platform, an application to control volume usage, and the SRC-ACP (Admission Control Plug-In) application.

Table 3: Juniper Networks C-series and SRC Technical Publications (continued)

Document	Description
Release Notes	
SRC-PE Release Notes *SRC Application Library Release Notes*	In the *Release Notes*, you will find the latest information about features, changes, known problems, resolved problems, supported platforms and network devices (such as Juniper Networks routers and CMTS devices), and third-party software. If the information in the *Release Notes* differs from the information found in the documentation set, follow the *Release Notes*. Release notes are included in the corresponding software distribution and are available on the Web.

Obtaining Documentation

To obtain the most current version of all Juniper Networks technical documentation, see the products documentation page on the Juniper Networks Web site at

http://www.juniper.net/

To order printed copies of this manual and other Juniper Networks technical documents or to order a documentation CD, which contains this manual, contact your sales representative.

Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation to better meet your needs. You can send your comments to techpubs-comments@juniper.net, or fill out the documentation feedback form at

http://www.juniper.net/techpubs/docbug/docbugreport.html

If you are using e-mail, be sure to include the following information with your comments:

- Document name
- Document part number
- Page number
- Software release version

Requesting Support

For technical support, open a support case using the Case Manager link at

http://www.juniper.net/support/

or call 1-888-314-JTAC (from the United States, Canada, or Mexico) or 1-408-745-9500 (from elsewhere).

Chapter 1

Managing Tiered and Premium Services with QoS on JUNOSe Routers

This chapter describes how to use the SRC software to manage QoS services that are available on JUNOSe routers. Topic include:

- Overview of QoS on JUNOSe Routers on page 1
- Dynamically Managing QoS Profiles on page 2
- Configuring QoS Profile-Tracking Plug-Ins with the SRC CLI on page 7
- Configuring QoS Profile-Tracking Plug-Ins with SDX Configuration Editor on page 10
- Updating QoS Profile Data in the Directory on page 13
- Searching for QoS Policy Data in the Directory on page 16

Overview of QoS on JUNOSe Routers

Tiered Internet access and premium services such as video on demand, gaming, or videoconferencing require QoS profiles to be running on the subscriber interface on the JUNOSe router. The router allows only one QoS profile to be attached to an interface at one time. Therefore, as a subscriber activates and deactivates different services, the QoS profile running on the interface needs to change. Also, as subscribers activate services, they may have multiple QoS services running at the same time; for example, internet-gold with videoconferencing.

With the SRC software, you can:

- Dynamically manage QoS profiles on the JUNOSe router to control a combination of services that require QoS.
- Update the directory and SDX Admin with a list of QoS profiles that are currently configured on a JUNOSe router.
- Search the directory for QoS policy information.

Dynamically Managing QoS Profiles

The SAE provides a QoS-tracking plug-in (QTP) that you can use to ensure that, as a subscriber activates and deactivates services, the required QoS profile is attached to the subscriber interface. With the QTP, the QoS profile selected is based on the activation state of an aggregation of services, not just one service.

For example, a subscriber activates a QoS service on a subscriber interface that requires a QoS profile that supports 512 best effort. The subscriber then activates a faster service (for example, 1024 best effort), as well as video on demand, and now has two QoS services running on an interface. The subscriber now needs a QoS profile to be attached to the interface that supports both video on demand and 1024 best-effort service. The QTP can determine which QoS profile the subscriber needs, and can cause the existing QoS profile to be removed from the subscriber interface and the new QoS profile to be attached to the interface.

Note that if a profile is installed on a subscriber interface and the QTP installs a new profile, the new profile is based on QoS services that are currently active. The new profile does not combine the functionality of the previous profile with the new profile. For example, if a subscriber has a default policy with QoS profile be-512 installed on the subscriber interface, and the subscriber activates a video-on-demand service, the QTP does not combine the functionality of be-512 with the profile that supports video on demand.

How QoS Profile Tracking Works

The SAE manages policies on router interfaces through service sessions. Service session configurations contain the policy that needs to be installed on an interface when a service is activated. The policy definition can include the name of a QoS profile to attach to the interface when the policy is installed.

When you set up the QTP, you create a QoS profile attachment service. The purpose of this service is to attach the required QoS profile to an interface. This service is hidden from subscribers and is under only QTP control.

Because profiles need to be changed only when QoS services are activated or deactivated, the QTP tracks services and reacts to service state changes by adjusting the QoS profile attachment as needed by deactivating and activating the QoS profile attachment service.

Subscribers who need their services managed by the QTP are subscribed to the QoS profile attachment service.

Identifying QoS Services

When you set up a service (SSP service) in SDX Admin, you identify the service as a QoS service in one of the fields in the service definition. For example, you can assign a service name or category to indicate that the service is a QoS service, or you could assign the QTP instance name in the Tracking Plugin field.

SDX Admin saves services in the sspService object class in the directory. When the SAE notifies the QTP that a service has been activated or deactivated, the QTP determines whether it is a QoS service by searching attributes in the service object. The QTP uses a search filter that you set up to search an attribute for the information that you assigned to the service to indicate that it is a QoS service.

For example, suppose you enter myqtp in the tracking plug-in field of QoS services to indicate that the service is a QoS service. You would set up the search filter to search tracking plug-in attributes for any service that contains myqtp:

```
(attribute.trackPlug=*myqtp*)
```

Or you might configure the category to indicate that a service is a QoS service. The following filter searches service category attributes for any entry that contains ultra, video on demand, or video telephony:

```
(|(serviceCategory=*ultra*)(|(serviceCategory=*video on
demand*)(serviceCategory=*video telephony*)))
```

To obtain a list of attribute names for the sspService object class, see the LDAP schema documentation in the SRC software distribution in the folder *SDK/doc/ldap* or on the Juniper Networks Web site at

http://www.juniper.net/techpubs/software/management/sdx

Determining the QoS Profile

After the QTP determines that a service is a QoS service, it needs to obtain the name of the QoS profile for the service. The QTP generates a QoS profile name based on active QoS services as follows:

1. Obtains QoS profile input values.

   The QTP obtains these values by taking the value of an attribute in the service definition. You specify which attribute that you want the QTP to use as the input value. For example, you can specify the service name, the category, or the contents of the design and graphics attribute.

2. Compiles a list of the QoS profile input values.

3. Removes duplicate values from the list.

4. Sorts the remaining list by using a case-sensitive alphanumeric comparison.

5. Concatenates the values with a separator. The default value for the separator is a hyphen (-). You can specify a different separator.

   Table 4 shows how lists of QoS profile input values are sorted and then concatenated.

Table 4: Examples of Concatenated QoS Profile Input Values

Input – QoS Profile Input Values	Output – Concatenated Name
be512, vod	be512-vod
game, be1024, vod	be1024-game-vod
be128	be128

6. Adds a prefix to the resulting name. The default prefix is qos-profile. (You can specify a different value.) The output from our examples in Table 4 now looks like this:

   - qos-profile-be512-vod
   - qos-profile-be1024-game-vod
   - qos-profile-be128

The names that result from this process are the QoS profile names.

As you can see from this process, you need to design services and configure the QTP so that the resulting QoS profile names match the names of the QoS profiles configured on the JUNOSe router.

Typically, a QoS designer creates a number of QoS profiles that support all the services that are expected to be used. This design results in various QoS profiles that need to be configured on each router. If a required QoS profile is not configured on the router, the hidden QoS profile attachment service cannot be activated. Services are still activated for the subscriber, but the services will not provide the expected traffic requirements. When this happens, the SAE logs the error but does not send an error message to the subscriber.

Setting Up Policy Groups

You need to create two types of policy groups in your QTP configuration. The QoS profile attachment service needs a policy group that attaches the required QoS profile to the subscriber interface when the attachment service is activated. QoS services need policy groups that classify traffic and specify the action to take on traffic that matches the classifier. (You can set up traffic classifiers to match any traffic.)

Policy Group for QoS Profile Attachment Service

The policy group for the hidden QoS profile attachment service must have an egress policy list with only one policy rule that contains a QoS profile attachment action. The QoS profile attachment action must have a variable parameter in the QoS profile field. The policy rule can also contain a classify traffic condition.

NOTE: The policy group for the QoS profile attachment service must contain only one egress policy list and must contain one and only one QoS profile attachment action. Otherwise, the SRC software will require a license for the hidden service.

When the profile attachment service is activated, the QTP substitutes the QoS profile attribute in the policy with the QoS profile name that it determined, as described in *Determining the QoS Profile* on page 3. The service then loads the policy.

The following example creates a policy group for the QoS profile attachment service. This policy group does not match any traffic.

1. Create a policy group called Pg-qos-attach, and add an egress policy list.

2. In the egress policy list, create a policy rule that has a classify-traffic condition that will not match any real traffic. For example, set both the source and destination addresses to 0.0.0.0/32.

3. In the egress policy list, create a policy rule that has a QoS profile attachment action with QoS profile qpName.

   By default, the QTP looks for qpName as the variable parameter.

   When the QTP determines the required QoS profile name, it substitutes qpName with the value that it acquired.

Setting Up Services

You need to set up a QoS profile attachment service and QoS services. Both types of services are value-added (SSP) services.

In the QoS profile attachment service, assign the policy group that you configured for the service. For example, policyGroupName = Pg-qos-attach, ou = ent, o = Policies, o = umc.

In QoS services, assign the policy group that you configured for the service.

Subscribe subscribers to the QoS profile attachment service and to the appropriate QoS services.

Reestablishing Default QoS Profile

A default QoS profile may be installed on the subscriber interface before the QTP installs QoS profiles in response to the activation of QoS services. For example, a profile may have been attached to the subscriber interface when the default policy was installed. Once QoS services are no longer active on the interface, the QTP can reestablish the QoS profile that was installed on the interface before the QTP began tracking services and installing profiles on the interface.

Example: How QTP Activates a QoS Service

The following example shows the process that QTP uses when a subscriber activates a QoS service. In this example, QoS profile input values are taken from the service name attribute. The hidden QoS profile attachment service is named svc-qos-attach. The svc-qos-attach service contains a policy that has the variable parameter qpName assigned as the QoS profile name.

1. The subscriber does not have any active services.

2. The subscriber activates service be512, which is a QoS service.

   a. The SAE sends a Service Session Start event to the QTP.

   b. The QTP searches an attribute in the service definition and determines that the service is a QoS service.

c. Using the SAE Common Object Request Broker Architecture (CORBA) remote application programming interface (API), the QTP gets a list of the subscriber's active QoS services.

   The list contains only service be512 because that is the only service that the subscriber has activated.

d. The QTP adds the default prefix to the QoS profile input value to obtain the QoS profile name. The result is:

   qos-profile-be512

e. The QTP deactivates the hidden svc-qos-attach service. Because this svc-qos-attach service was not active before, this operation does not have any effect.

f. The QTP activates the hidden svc-qos-attach service, and it substitutes variable parameter qpName with $'qos-profile-be512' as the QoS profile name in the policy.

g. The policy loads qos-profile-be512 on the subscriber interface.

3. The subscriber activates service vod, which is a QoS service.

   a. The SAE sends a Service Session Start event to the QTP.

   b. QTP searches attributes in active service definitions and determines that the service is a QoS service.

   c. The QTP gets a list of the subscriber's active QoS services. The result is:

      be512, vod

   d. The QTP sorts the list and concatenates the QoS profile input values with the separator. The result is:

      be512-vod

   e. The QTP adds the default prefix to the concatenated name to obtain the QoS profile name. The result is:

      qos-profile-be512-vod.

   f. The QTP deactivates the hidden svc-qos-attach service.

   g. The QTP activates the hidden svc-qos-attach service, and it substitutes variable parameter qpName with $'qos-profile-be512-vod' as the QoS profile name in the policy.

   h. The policy loads qos-profile-be512-vod.

4. The subscriber deactivates service vod.

   a. The QTP follows the same procedure as in Step 2 above and determines that the QoS profile name is qos-profile-vod.

   b. The QTP deactivates the hidden svc-qos-attach service.

   c. The QTP reactivates the hidden svc-qos-attach service, and it substitutes variable parameter qpName with $'qos-profile-be512' as the QoS profile name in the policy.

   d. The policy loads qos-profile-be512.

Configuring QoS Profile-Tracking Plug-Ins with the SRC CLI

Use the following configuration statements to configure the QoS profile tracking plug-in with the SRC CLI:

```
shared sae configuration plug-ins pool name qos-profile-tracking {
    threads threads;
    default-qos-profile default-qos-profile;
    separator separator;
    qos-profile-prefix qos-profile-prefix;
    service-selection-attribute service-selection-attribute;
    search-filter search-filter;
    invisible-qos-service invisible-qos-service;
    qos-profile-parameter-name qos-profile-parameter-name;
}
```

1. From configuration mode for the QoS profile tracking plug-in.

   ```
   user@host# edit shared sae configuration plug-ins pool QosTracking
   qos-profile-tracking
   ```

2. Configure the number of working threads that all QTP instances share when they process QTP events.

   ```
   [edit shared sae configuration plug-ins pool QosTracking qos-profile-tracking]
   user@host# set threads threads
   ```

3. Configure the name of the QoS profile that is attached to the interface when QoS services have been deactivated.

   See *Reestablishing Default QoS Profile* on page 5.

   ```
   [edit shared sae configuration plug-ins pool QosTracking qos-profile-tracking]
   user@host# set default-qos-profile default-qos-profile
   ```

4. Configure the character that is placed between QoS profile input values when the system concatenates the values during the process of creating QoS profile names.

   ```
   [edit shared sae configuration plug-ins pool QosTracking qos-profile-tracking]
   user@host# set separator separator
   ```

5. Configure the prefix added to the QoS service name as part of the process to determine the name of the QoS profile that needs to be attached to an interface for a particular service.

   [edit shared sae configuration plug-ins pool QosTracking qos-profile-tracking]
   user@host# **set qos-profile-prefix** *qos-profile-prefix*

6. Configure the name of the attribute in the service definition that you want the QTP to use as QoS profile input values.

   [edit shared sae configuration plug-ins pool QosTracking qos-profile-tracking]
   user@host# **set service-selection-attribute** *service-selection-attribute*

7. Configure the search filter that the SAE uses to search service objects in the directory to find QoS services.

   See *Configuring Search Filters for QoS Profile-Tracking Plug-Ins* on page 9

   [edit shared sae configuration plug-ins pool QosTracking qos-profile-tracking]
   user@host# **set search-filter** *search-filter*

8. Configure the name of the hidden QoS profile attachment service that the QTP uses to attach QoS profiles to and remove QoS profiles from a router interface.

   [edit shared sae configuration plug-ins pool QosTracking qos-profile-tracking]
   user@host# **set invisible-qos-service** *invisible-qos-service*

9. Configure the name of the variable parameter used in the QoS profile name field in the QoS profile attachment action of the policy group that is assigned to the hidden QoS service.

   [edit shared sae configuration plug-ins pool QosTracking qos-profile-tracking]
   user@host# **set qos-profile-parameter-name** *qos-profile-parameter-name*

10. Verify your configuration.

```
[edit shared sae configuration plug-ins pool QosTracking
qos-profile-tracking]
user@host# show
threads 1;
default-qos-profile ;
separator -;
qos-profile-prefix qos-profile;
service-selection-attribute serviceName;
search-filter (attribute.trackPlug=);
invisible-qos-service svc-qos-attach;
qos-profile-parameter-name qpName;
```

Configuring Search Filters for QoS Profile-Tracking Plug-Ins

The SAE uses a search filter to search service objects in the directory to find QoS services. You can set up the filter to search the values of any attribute in the service object, such as service name, category, or tracking plug-in. The search is successful when a value matches the filter.

For information about obtaining a list of attribute names for the sspService object class, see the documentation for the LDAP schema in the SRC software distribution in the folder *SDK/doc/ldap* or on the Juniper Networks Web site at

http://www.juniper.net/techpubs/software/management/sdx

Configure the search filter in a format similar to the LDAP search filter. Table 5 lists the values that you can use for filters. Each filter string <filter> contains a simplified LDAP query.

Table 5: Settings for Filter Strings

Filter String	Action
()	Matches no objects
(*)	Matches all objects
List of <attribute> = <value> pairs <attribute>—Name of a property or attribute <ldapAttributeName> <value>—One of the following ■ * (asterisk) ■ Explicit string ■ String that contains an * **Note:** To define a special character (* & , ! | \ )in a string, precede it with the backslash symbol (\).	■ If <value> is *, checks for any value. ■ If <value> is an explicit string, checks whether any value of the property matches the string, regardless of case. ■ If <value> is a string that contains a *, checks whether any value of the property contains the string, regardless of case.
(& <filter> <filter> ...)	True if all filters match
(| <filter> <filter> ...)	True if at least one filter matches
(! <filter>)	True if the filter does not match

- Default—(attribute.trackPlug=); note that you need to add a search value after the equal sign
- Examples
  - To search tracking plug-in attributes for any entry that contains qtp:

    (attribute.trackPlug=*qtp*)

  - To search service category attributes for any entry that contains ultra, video on demand, or video telephony:

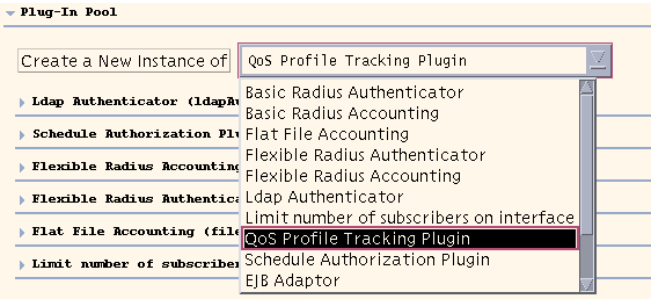
    (|(serviceCategory=*ultra*)(|(serviceCategory=*video on demand*)(serviceCategory=*video telephony*)))

Configuring QoS Profile-Tracking Plug-Ins with SDX Configuration Editor

To create a QoS profile tracking plug-in with SDX Configuration Editor:

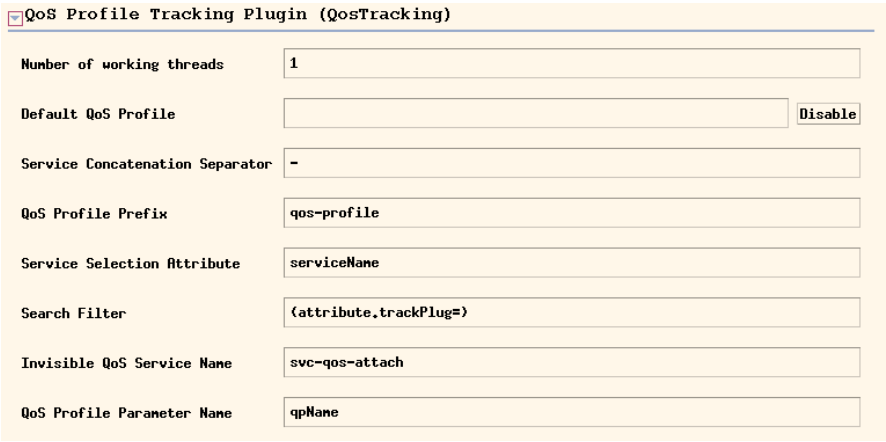
1. In the navigation pane, select the SAE object for which you want to configure plug-ins.
2. Click the Plug-Ins tab.

   The Plug-Ins pane appears.
3. Expand Plug-In Pool. From the drop-down list next to the Create a New Instance of button, select **QoS Profile Tracking Plugin**, and click **Create a New Instance of**.



   The Create New Instance dialog box appears.
4. Assign a name to the instance, and click **OK**.

   The instance appears in the plug-in pool.



5. Fill in the fields for the plug-in instance as described below.

Number of QoS Working Threads

- Number of working threads that all QTP instances share when they process QTP events.
- Value—Integer in the range 1–100
- Default—1
- Property name—NumQosTrackingThreads

Default QoS Profile

- Name of the QoS profile that is attached to the interface when QoS services have been deactivated. See *Reestablishing Default QoS Profile* on page 5.
- Value—Name of QoS profile
- Default—No value
- Property name—noServiceQosProfile

Service Concatenation Separator

- Character that is placed between QoS profile input values when the system concatenates the values during the process of creating QoS profile names. See Table 4 on page 3.
- Value—Any character that is valid in QoS profile names on the router
- Default—A single hyphen (-)
- Property name—NameSeparator

QoS Profile Prefix

- Prefix added to the QoS service name as part of the process to determine the name of the QoS profile that needs to be attached to an interface for a particular service. See *Determining the QoS Profile* on page 3.
- Value—Prefix that, when combined with QoS profile input values, matches a QoS profile on the router
- Default—qos-profile
- Property name—qosProfilePrefix

Service Selection Attribute

- Name of the attribute in the service definition that you want the QTP to use as QoS profile input values. The QTP uses these values to determine the name of the QoS profile that needs to be attached to an interface for a group of QoS services.
- For information about obtaining a list of attribute names for the sspService object class, see the documentation for the LDAP schema in the SRC software distribution in the folder *SDK/doc/ldap* or on the Juniper Networks Web site at

  http://www.juniper.net/techpubs/software/management/sdx

- Value—Name of any attribute in the service object; for example, serviceCategory, sspDesignAndGraphics

- Default—serviceName
- Property name—serviceSelectAttributes

Search Filter

- Search filter that the SAE uses to search service objects in the directory to find QoS services. You can set up the filter to search the values of any attribute in the service object, such as service name, category, or tracking plug-in. The search is successful when a value matches the filter.

  For information about obtaining a list of attribute names for the sspService object class, see the documentation for the LDAP schema in the SRC software distribution in the folder *SDK/doc/ldap* or on the Juniper Networks Web site at

  http://www.juniper.net/techpubs/software/management/sdx

- Value—Search filter in a format similar to the LDAP search filter. Table 6 lists the values that you can use for filters. Each filter string <filter> contains a simplified LDAP query.

Table 6: Settings for Filter Strings

Filter String	Action
()	Matches no objects
(*)	Matches all objects
List of <attribute> = <value> pairs <attribute>—Name of a property or attribute <ldapAttributeName> <value>—One of the following ■ * (asterisk) ■ Explicit string ■ String that contains an * **Note:** To define a special character (* & , ! | \ )in a string, precede it with the backslash symbol (\).	■ If <value> is *, checks for any value. ■ If <value> is an explicit string, checks whether any value of the property matches the string, regardless of case. ■ If <value> is a string that contains a *, checks whether any value of the property contains the string, regardless of case.
(& <filter> <filter> ...)	True if all filters match
(| <filter> <filter> ...)	True if at least one filter matches
(! <filter>)	True if the filter does not match

- Default—(attribute.trackPlug=); note that you need to add a search value after the equal sign

- Examples
  - To search tracking plug-in attributes for any entry that contains qtp:

    (attribute.trackPlug=*qtp*)

  - To search service category attributes for any entry that contains ultra, video on demand, or video telephony:

    (|(serviceCategory=*ultra*)(|(serviceCategory=*video on demand*)(serviceCategory=*video telephony*)))

- Property name—ServiceSelectFilter

Invisible QoS Service Name

- Name of the hidden QoS profile attachment service that the QTP uses to attach QoS profiles to and remove QoS profiles from a router interface.
- Value—Name of the configured service
- Default—svc-qos-attach
- Property name—InvisibleQosServiceName

QoS Profile Parameter Name

- Name of the variable parameter used in the QoS profile name field in the QoS profile attachment action of the policy group that is assigned to the hidden QoS service. When the QTP obtains the name of the required QoS profile, it substitutes that value for the variable parameter.
- Value—Valid parameter name
- Default—qpName
- Property name—qosParameterName

Updating QoS Profile Data in the Directory

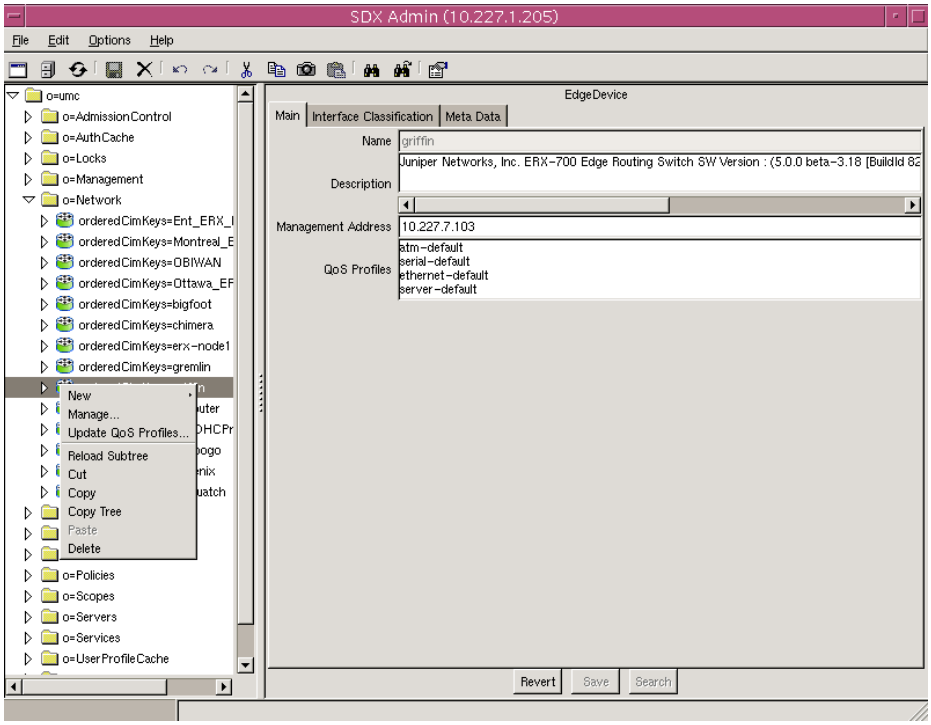
You can update the directory and SDX Admin with a list of QoS profiles that are currently configured on a JUNOSe router. You can do so by using either SDX Admin or a program called qosProfilePublish.

Note that this feature is not supported on the C-series platform.

Using SDX Admin to Update QoS Profile Data

To update the directory with SDX Admin:

1. In the navigation pane, expand the object *o = Network*.
2. Select the router for which you want to update QoS profiles, and right-click.



3. Select Update QoS Profiles.

   The SDX Admin dialog box appears.

4. Enter the IP address for the router; enter the SNMP community if the default value is incorrect; and click **OK**.

   SDX Admin updates the QoS profiles for the router in the directory and displays the information in the QoS Profiles field of the Main tab in the EdgeDevice pane.

Using qosProfilePublish to Update QoS Profile Data

Because QoS profiles are part of the global configuration of JUNOSe routers, when a QoS profile is configured on the router, all virtual routers (VRs) can use that profile. Therefore, you update QoS profiles per router, not per VR as you do with IP pools. However, when you run qosProfilePublish, you still must define a VR using the **-v** option.

The syntax for qosProfilePublish is:

qosProfilePublish { { -v <vrName> @ <routerName> -i <ipAddress> }*
-h <host> -b <baseDn> -D <bindDN> -w <password>
-c <readCommunity>] | -H }

To update QoS profile data using the **qosProfilePublish** command:

1. On the SAE host, access the folder */opt/UMC/sae/etc*.

   cd /opt/UMC/sae/etc

2. Run the command.

   ./qosProfile -v vr1@erx1 -i 192.0.2.1 -v vr2@erx2 -i 192.0.2.3 -h 192.0.2.5 -w admin123 -D cn=umcAdmin,o=umc -b o=Network,o=umc -c public

<vrName>

- Name of the VR.
- Value—Text string (value is case sensitive and must match the name in the JUNOSe configuration)
- Example—vr-boston

<routerName>

- Name of the JUNOSe router from which you want to update QoS profiles.
- Value—Text string (value is case sensitive and must match the name in the JUNOSe configuration)
- Example—erx1

<ipAddress>

- JUNOSe router IP address.
- Value—IP address or text string
- Example—192.0.2.1

<host>

- IP address or name of the host that supports the directory.
- Value—IP address or text string
- Example—192.0.2.2 or ottawa

<baseDn>

- DN of the root of the tree in the directory.
- Value—DN
- Example—*o = Network,o = umc*

<bindDn>

- DN of the username for authentication with the directory server.
- Value—DN
- Example—*cn = umcAdmin,o = umc*

<password>

- Password for authentication with the directory server.
- Value—Text string
- Example—admin123

<readCommunity>

- Name of the SNMP read community for a VR. If the SNMP read community for a VR is defined in the directory, you do not need to specify this value.
- Value—Text string
- Example—Public

-H

- Online help for this tool.

To update QoS profiles with qosProfilePublish:

1. Access the folder in which qosProfilePublish is installed.

   cd /opt/UMC/sae/etc

2. Run qosProfilePublish.

   The program accesses QoS profiles for the router that you specify and updates the information in the specified directory.

   ```
   # ./qosProfilePublish -v default@erx1 -i 10.10.7.28 -h 10.10.227.7 -w admin123
   -D cn=umcAdmin,o=umc -b o=Network,o=umc -c public
   erx1 profiles are: [‘atm-default’, ‘serial-default’,
   ‘ethernet-default’, ‘server-default’]
   ```

Searching for QoS Policy Data in the Directory

Note that this feature is not supported on the C-series platform.

You can run queries of the directory data to find:

- QoS profiles configured on a JUNOSe router.
- QoS profiles in a policy group.
- Policy groups that contain a particular QoS profile.
- JUNOSe routers that have a QoS profile configured.

- Policy groups supported on a router. For a policy group to be supported on a router, both the policy group and the router must contain the same QoS profile.
- Routers that can be supported by a policy group. The query provides a list of routers that contain QoS profile(s) that are also in the specified policy group.

You can run these queries by using either Policy Editor or Policy Web Admin.

Using Policy Editor to Search for QoS Policy Information

Before using Policy Editor to run a query, you need to:

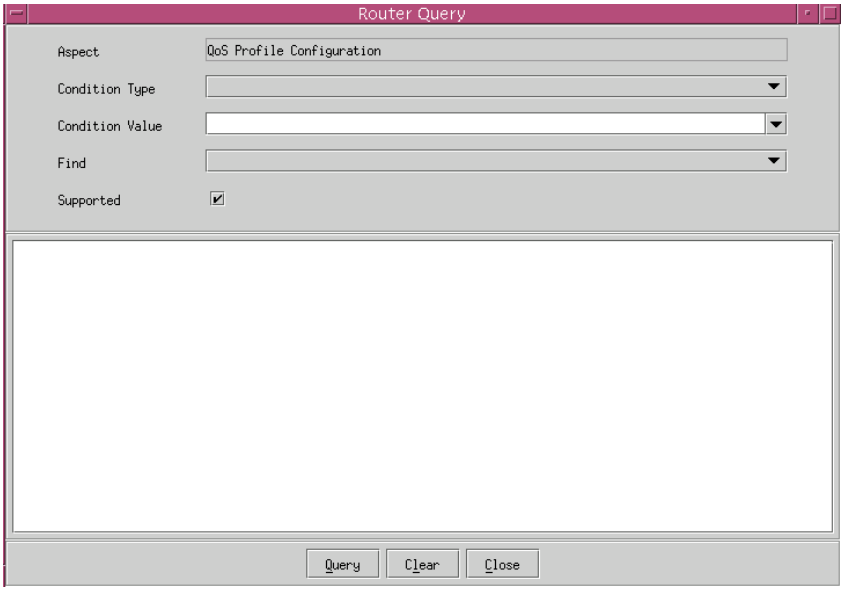
- Connect Policy Editor to a directory server. See *Starting Policy Editor* in *SDX Services and Policies Guide, Chapter 7, Using Policy Editor*.
- Update the directory with a list of QoS profiles that are on the router(s) that you want to search. See *Updating QoS Profile Data in the Directory* on page 13.

Running Queries from Policy Editor

To run queries with Policy Editor:

1. In the Policy Editor window, click Tools in the menu bar; then click Query.

   The Router Query window appears.



2. Fill in the fields, and click **Query**.

To erase query results from the screen, click **Clear**.

Condition Type

- Object to be searched.
- Value—router, QoS profile, or policy group
- Default—No value

Condition Value

- Name of the QoS profile, router, or policy group that you want to search.
- Value—Name of the router, QoS profile, or policy group. If you selected router or policy group as a condition type, you can select a name from the drop-down menu. If the condition type is QoS profile, continue selecting entries in the drop-down menu until you reach the name of a policy group.
- Default—No value

Find

- Object that you want to find. The software searches for this object on the QoS profile, router, or policy group defined in condition type and condition value.
- Value—router, QoS profile, or policy group
- Default—No value

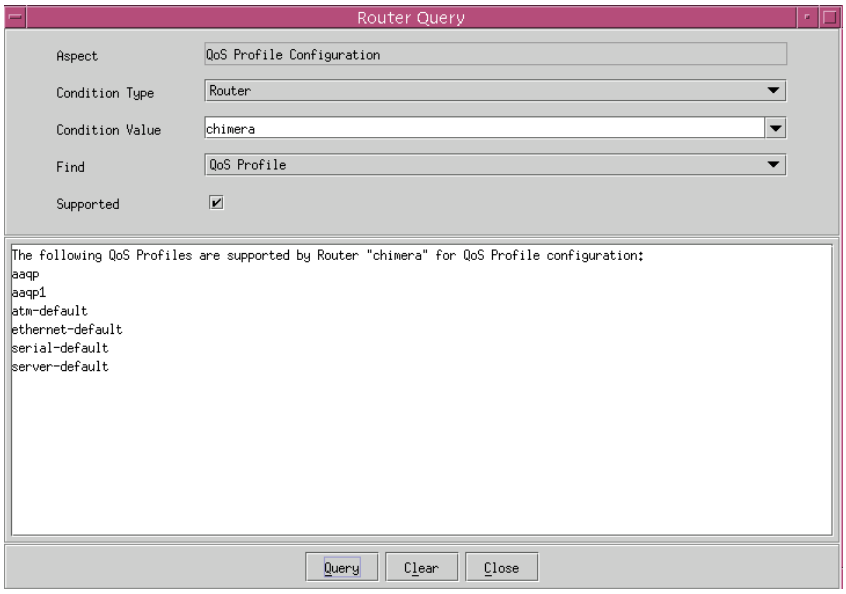
Supported

- Whether or not to search for the condition type that exists or does not exist on the router, QoS profile, or policy group.
- Value—Checked or unchecked
  - Checked—Searches for the condition type that is on the router, QoS profile, or policy group
  - Unchecked—Searches for the condition type that is not on the router, QoS profile, or policy group
- Default—No value

Examples

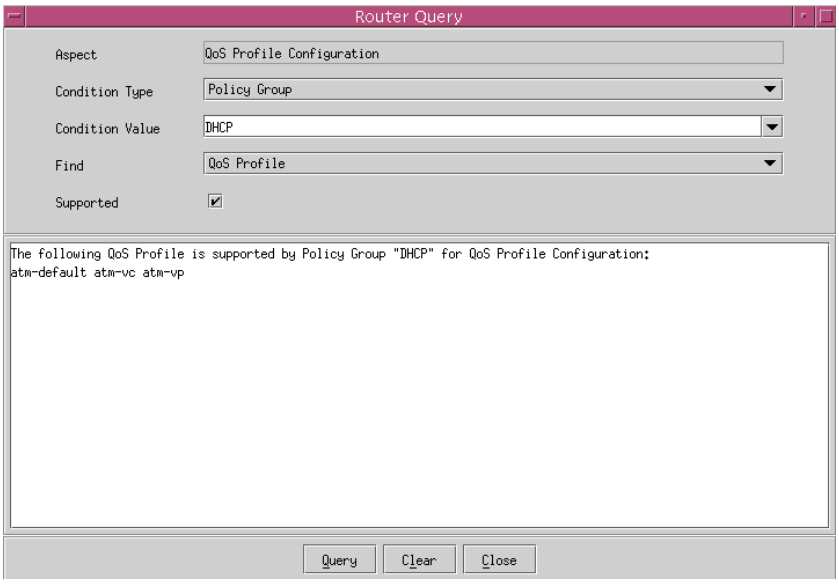
The query example in Figure 1 searches for all QoS profiles on router chimera.

Figure 1: Searching for All QoS Profiles on a Router



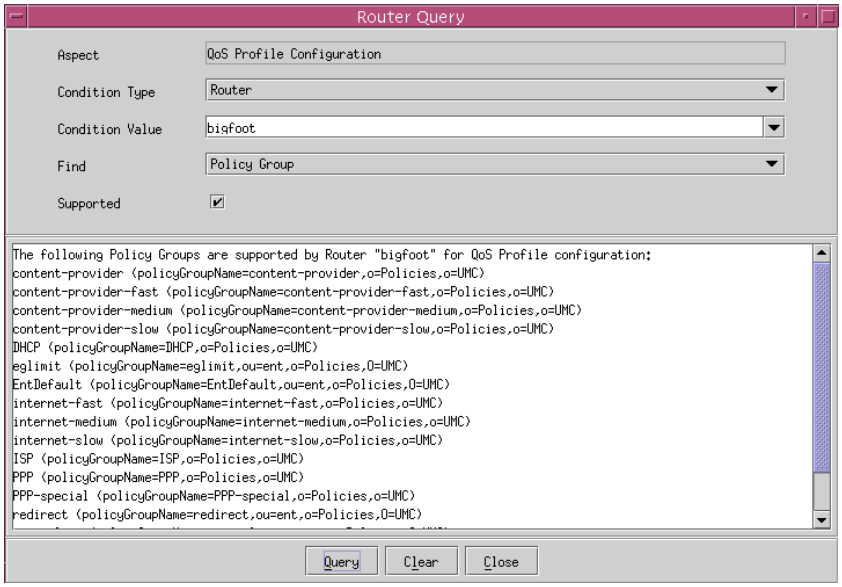
The query in Figure 2 searches for QoS profiles in policy group DHCP.

Figure 2: Searching for QoS Profiles in a Policy Group



The query in Figure 3 searches for all policy groups that router bigfoot supports. For a policy group to be supported on a router, both the policy group and the router must contain the same QoS profile.

Figure 3: Searching for All Policy Groups on a Router



Using Policy Web Admin to Search for QoS Policy Information

Before you use Policy Web Admin, deploy the WAR file for the Policy Web Admin in the Web application server. You can find this file, *pomAdmin.war*, in the folder *webapp* on the SRC software distribution. Refer to the documentation for the Web application server for information about deploying applications.

To deploy Policy Web Admin inside JBoss:

- Copy the file to the JBoss *server/default/deploy* directory.

  cp /cdrom/cdrom0/webapp/pomAdmin.war
  /opt/UMC/jboss/server/default/deploy

  JBoss automatically starts the application when a WAR file is copied into the deploy directory.

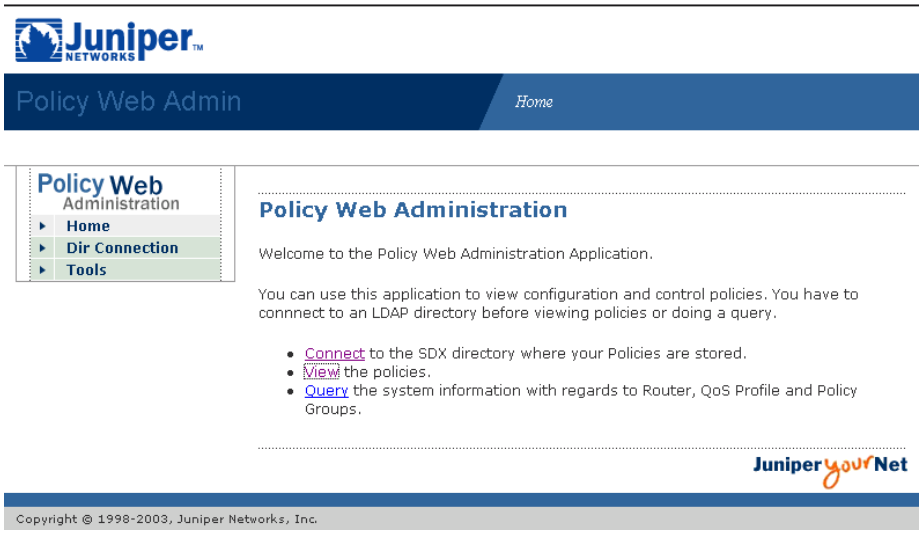
Launching Policy Web Admin

To launch Policy Web Admin:

1. Start your Web browser.
2. Enter the location of Policy Web Admin in the following format:

   https://<web-server-name or ip-address>:<port>/pomAdmin

   The Policy Web Admin page appears.

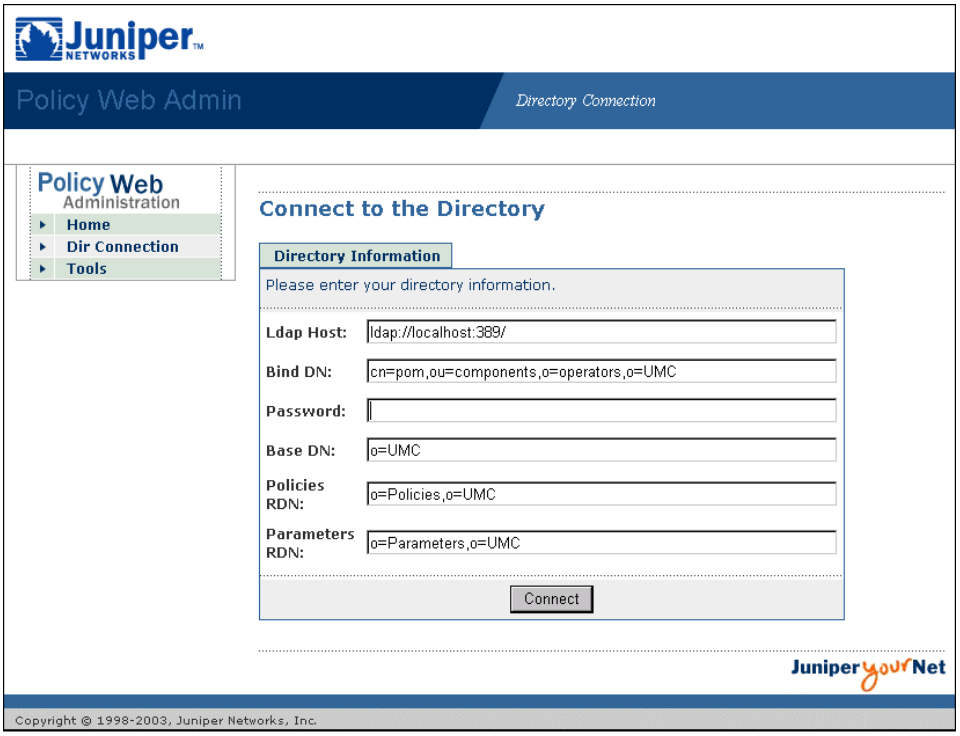


Connecting to a Directory

Before you run queries, you need to connect to the directory where policies are stored. To connect to the directory:

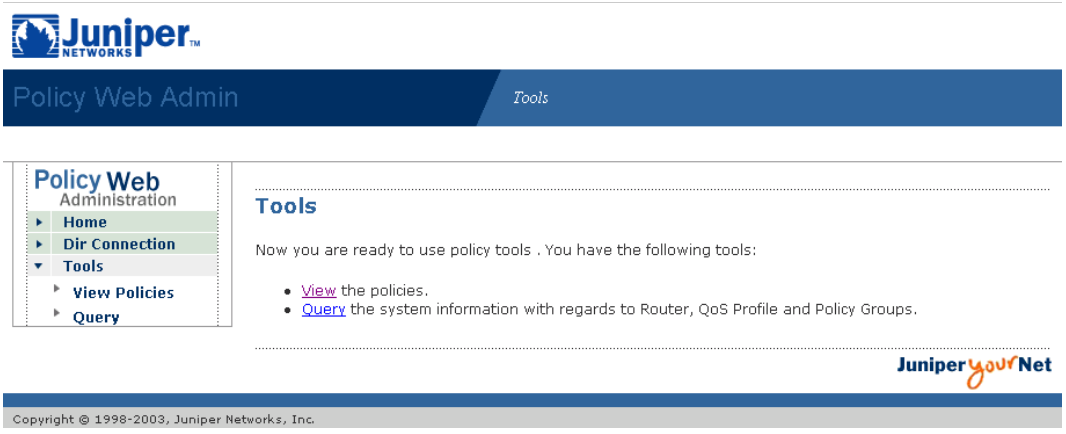
1. From the Policy Web Admin main window, click **Dir Connection**.

   The Directory Connection page appears.



2. Enter the connection information for the directory that contains the policies, and click **Connect**.

   The Tools page appears.

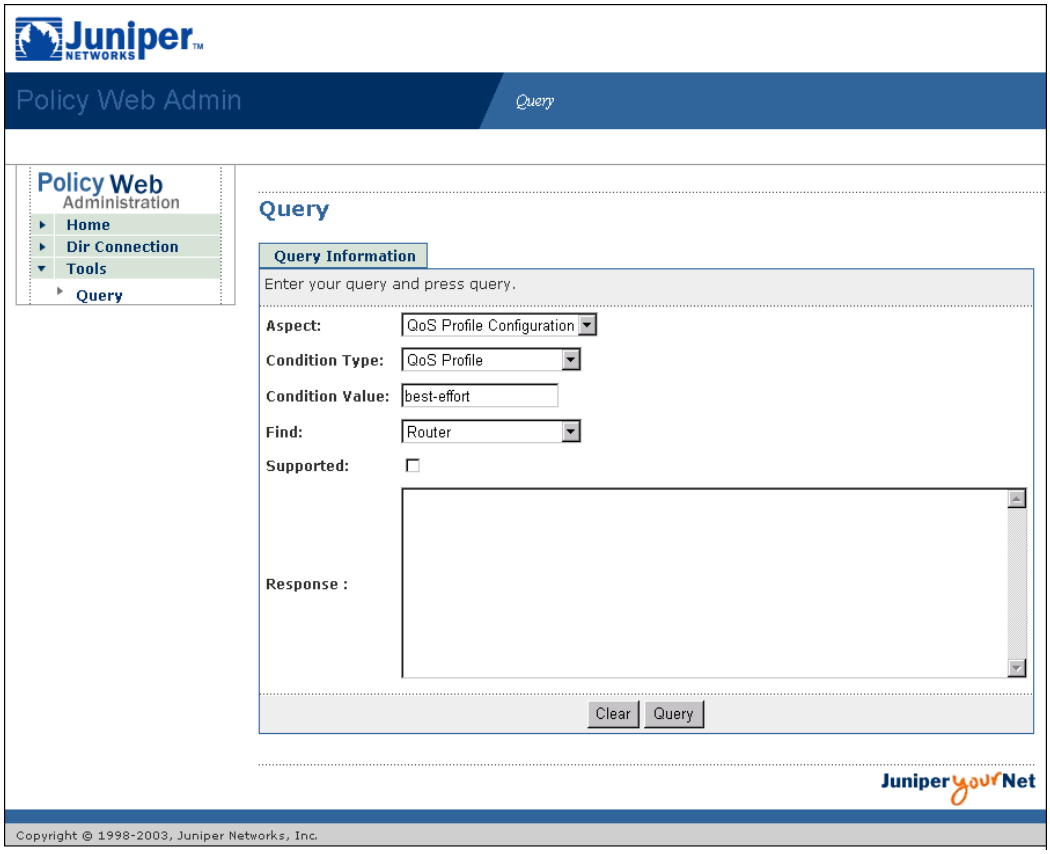


Querying the Directory for QoS Information

To search the directory for QoS information:

1. In the Tools page, click **Query**.

   The Query page appears.



2. Fill in the parameters, and click **Query**.

   The results appear in the Response field.

For examples of queries, see *Examples* on page 19.

Chapter 2

Managing Subscribers for a Wireless Roaming Environment

This chapter describes how you can use the SAE to manage wireless locations that support roaming from one wireless location to another. The chapter contains the following sections:

- Overview of a Wireless Roaming Environment on page 25
- Subscriber Access in a Wireless Roaming Environment on page 26
- Configuring Subscriber Access for a Wireless Location on page 27

Overview of a Wireless Roaming Environment

In a roaming wireless environment, subscribers can log in to a wireless access point at a variety of wireless locations owned by service providers that participate in a roaming network agreement. The wireless locations participating in the agreement can be owned by one or more service providers.

Typically, RADIUS manages information about subscribers between the wireless locations. A RADIUS server for an Internet service provider (ISP) manages authentication for its subscribers, and shares information with the other ISPs with which the service provider has a roaming agreement. Subscribers can log in to an SAE from any supported site.

The SAE provides support for RADIUS vendor-specific attributes for wireless Internet service provider roaming (WISPr). For more information about these attributes, see

http://www.wi-fialliance.org/opensection/wispr.asp

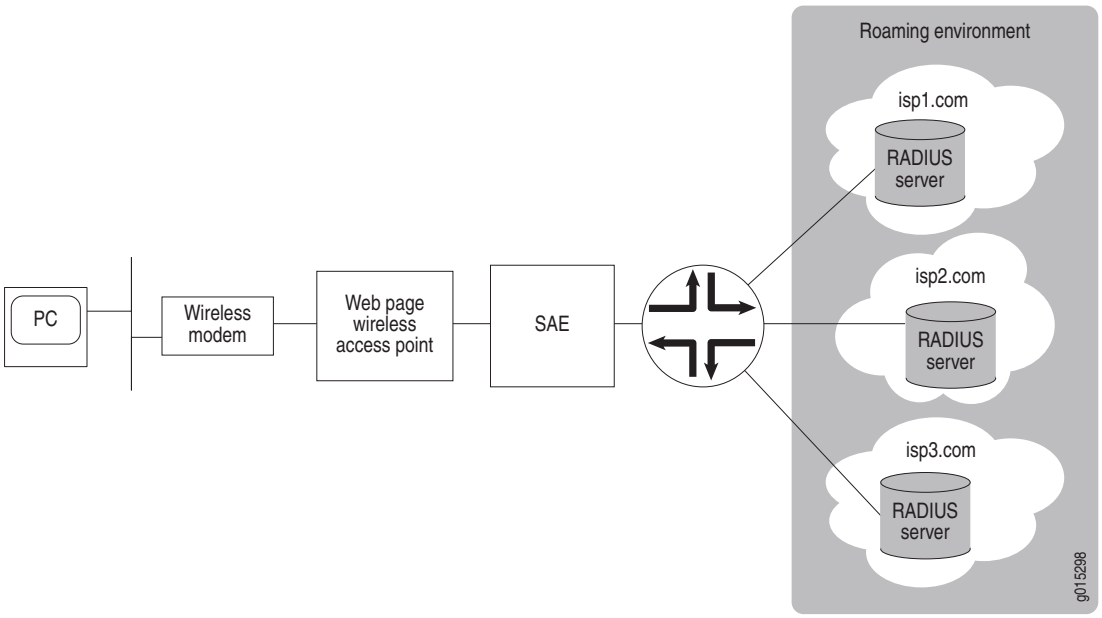
Subscriber Access in a Wireless Roaming Environment

When subscribers log in to a wireless location that has a roaming agreement with other locations, the following sequence of events occurs:

1. Subscribers connect to the local wireless location and provide login information on a portal page that provides a universal access method. This login information is forwarded to the SAE.
2. Based on the login information, an access service starts.
3. The subscriber is authenticated by RADIUS; the authorization includes RADIUS vendor-specific attributes for WISPr.
4. Policies are activated for the subscriber on the router.
5. After successful start of the access service, the portal page redirects the subscriber to a specified start page.

Figure 4 shows how subscribers interact with an SAE-managed wireless location that has a roaming agreement with wireless locations.

Figure 4: Subscriber Access to a Wireless Roaming Group



Configuring Subscriber Access for a Wireless Location

To use the SAE to manage a wireless access point that participates in a roaming agreement:

1. Configure RADIUS authentication for users who connect from a wireless location.
2. Create subscriber access to an ISP.
3. Create Web access.
4. Verify idle timeout properties for the SAE.

The following sections describe how to perform these tasks.

Configuring RADIUS Authentication

To set up RADIUS authentication to support a roaming environment between wireless Internet service providers, you can use the Flexible RADIUS Authentication plug-in that is provided with the SRC software, or you can create a custom RADIUS authentication plug-in.

Configuring a Custom RADIUS Authentication Plug-In

If you create a custom plug-in, be sure that it supports the same RADIUS attributes as those configured for the flexible RADIUS authentication plug-in. See *Configuring the Flexible RADIUS Authentication Plug-In* on page 27.

For information about creating a custom plug-in, see *SAE CORBA Plug-In Service Provider Interface (SPI)* in the SRC software distribution in the folder *SDK/doc/idl* or on the Juniper Networks Web site at

http://www.juniper.net/techpubs/software/management/sdx/api-index.html

Configuring the Flexible RADIUS Authentication Plug-In

The default flexible RADIUS authentication plug-in, flexRadiusAuth, provides support for RADIUS vendor-specific attributes for WISPr, which are listed in the following procedure. These attributes use the IANA private enterprise number 14122 assigned to the Wi-FI Alliance. For more information about these attributes, see

http://www.wi-fialliance.org/opensection/wispr.asp

You should be familiar with the general procedure for configuring the flexible RADIUS authentication plug-in before configuring it to include the WISPr attributes. For information about configuring the flexible RADIUS authentication plug-in, see *SDX Subscribers and Subscriptions Guide, Chapter 5, Configuring Subscriber-Related Properties on the SAE on a Solaris Platform*.

When you configure the plug-in, you can use the following standard attribute values to set values in authentication response packets:

- setAcctInterimTime
- SetSubstitution
- SetTerminateTime

Examples in the following procedure show how you can use these attribute values.

To configure the plug-in to support a roaming environment:

1. Configure attributes.
    - Required attributes:
        - An identifier for the wireless location:

          vendor-specific.WISPr.Location-ID=*Identifier*

          This attribute can be an interface description (ifAlias) or other value that identifies the JUNOSe interface to which the wireless access point connects.
        - The URL of the start page returned by the RADIUS server of the ISP:

          vendor-specific.WISPr.Redirection-URL=*Command to make the URL available to the SRC software*

          For example:

          vendor-specific.WISPr.Redirection-URL=setProperty(“startURL=%s” % ATTR)

          The default configuration sets a session property named startURL.
        - The URL of a page that a subscriber can use to log out of the network:

          vendor-specific.WISPr.Logoff-URL=*URL of a log out page*
    - Bandwidth attributes (recommended):
        - The maximum transmission rate in bites per second:

          vendor-specific.WISPr.Bandwidth-Max-Up=*Command to make the rate available to the SRC software*

          For example:

          vendor-specific.WISPr.Bandwidth-Max-Up=setSubstitution(“max_up_rate=%s” % ATTR)

- The maximum receive rate in bites per second:

  vendor-specific.WISPr.Bandwidth-Max-Down=*Command to make the rate available to the SRC software*

  For example:

  vendor-specific.WISPr.Bandwidth-Max-Down=setSubstitution(“max_down_rate=%s” % \ ATTR)

- Optional attributes:
  - The name of the wireless location:

    vendor-specific.WISPr.Location-Name=*Name of the wireless location*

  - The date and time that the subscriber session is to end:

    vendor-specific.WISPr.Session-Terminate-Time=*Command to set the session terminate time*

    For example:

    vendor-specific.WISPr.Session-Terminate-Time=setTerminateTime(ATTR)

  - The end of the subscriber session at the end of the billing day:

    vendor-specific.WISPr.Session-Terminate-End-Of-Day=ATTR or setTerminateTime("00:00:00")

    If the operator of the wireless location does not support daily billing, do not configure this attribute, and remove it if present.

  - A service type for billing:

    vendor-specific.WISPr.Billing-Class-Of-Service=*Service type*

2. For each attribute that you configure, configure the packet type to which the attribute applies. Table 7 shows the packet types associated with each attribute.

Table 7: Packet Types for RADIUS Attributes

RADIUS Attribute	Associated RADIUS Packet Definition
vendor-specific.WISPr.Location-ID	RadiusPacket.stdAuth.auth.vendor-specific.WISPr.Location-ID
vendor-specific.WISPr.Redirection-URL	RadiusPacket.stdAuth.auth.vendor-specific.WISPr.Redirection-URL
vendor-specific.WISPr.Logoff-URL	RadiusPacket.stdAuth.auth.vendor-specific.WISPr.Logoff-URL
vendor-specific.WISPr.Bandwidth-Max-Up	RadiusPacket.stdAuth.auth.vendor-specific.WISPr.Bandwidth-Max-Up
vendor-specific.WISPr.Maximum-Max-Down	RadiusPacket.stdAuth.auth.vendor-specific.WISPr.Maximum-Max-Down
vendor-specific.WISPr.Location-Name	RadiusPacket.stdAuth.auth.vendor-specific.WISPr.Location-Name
vendor-specific.WISPr.Session-Terminate-Time	RadiusPacket.stdAuth.auth.vendor-specific.WISPr.Session-Terminate-Time

Table 7: Packet Types for RADIUS Attributes

RADIUS Attribute	Associated RADIUS Packet Definition
vendor-specific.WISPr.Session-Terminate-End-Of-Day	RadiusPacket.stdAuth.auth.vendor-specific.WISPr.Session-Terminate-End-Of-Day
vendor-specific.WISPr.Billing-Class-Of-Service	RadiusPacket.stdAuth.auth.vendor-specific.WISPr.Billing-Class-Of-Service

Creating Subscriber Access to an ISP

An access service lets subscribers connect to an ISP. The policies associated with the access service should specify a JUNOS policing or JUNOSe rate-limiting policy to set the maximum bandwidth at which a subscriber can send traffic, and the maximum bandwidth at which a subscriber can receive traffic. When you configure the policies, define the bandwidth values as parameters so that the policies can be applied across a number of subscribers.

To configure an access service to the ISP:

1. In SDX Admin, create the access service.

   See *SDX Services and Policies Guide, Chapter 2, Managing Services on a Solaris Platform*.

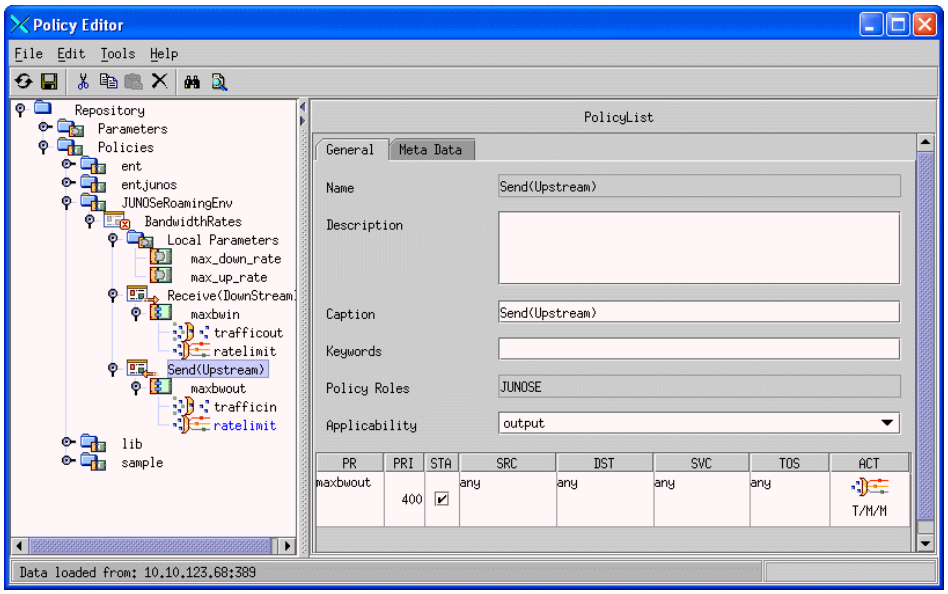
2. In Policy Editor, create a policy group the sets the maximum bandwidth at which a subscriber can send traffic, and the maximum bandwidth at which a subscriber can receive traffic. Use parameters to set these values.

   See *SDX Services and Policies Guide, Chapter 12, Configuring and Managing Policies with Policy Editor* and *SDX Services and Policies Guide, Chapter 15, Defining and Acquiring Values for Parameters*.

The example in Figure 5 on page 31 shows a policy configuration that includes:

- A local parameter named max_up_rate that sets the maximum rate at which the subscriber can send data
- A local parameter named max_down_rate that sets the maximum rate at which the subscriber can receive data
- A policy group Receive(Downstream) that references max_down_rate
- A policy group Send(Upstream) that references max_up_rate

Figure 5: Sample Rate-Limiting Policies with Bandwidth Parameters



Substitutions for these parameters can then be referenced in the RADIUS attributes:

```
vendor-specific.WISPr.Bandwidth-Max-Up=setSubstitution("max_up_rate=%s"
% ATTR)
vendor-specific.WISPr.Bandwidth-Max-Down=setSubstitution("max_down_rate=%s"
% ATTR)
```

Creating Web Access

When subscribers connect to and log in to a wireless access point, they are directed to a single Web page that is referred to as a captive portal page. This page is part of a residential service selection portal. A captive portal page receives and manages redirected Web requests. For information about residential portals and captive portal pages, see *SDX Subscribers and Subscriptions Guide, Chapter 15, Overview of the Residential Portal*.

When creating a captive portal page for a wireless roaming environment, configure the page to:

- Start an access service that is configured to be authenticated by the RADIUS server of the ISP.
- After the access service starts, redirect the subscriber to the page specified by the Redirect-URL RADIUS attribute. This page is the start page for the subscriber's home ISP.

  You can retrieve the URL of the start page from the service session property startURL. Note that startURL is the default name used for the flexible RADIUS authentication plug-in; you can assign a different name to this property.

  You can use the Subscriber.readSubscription() method in the Common Object Request Broker Architecture (CORBA) remote application programming interface (API) to retrieve the redirect URL.

Note that when you develop the portal, you can use the following methods in the SAE CORBA remote API to retrieve session data after the access service starts:

- Subscriber.readSubscriber()
- Subscriber.readSubscription()

For more information about these methods, see the SAE CORBA remote API documentation in the SRC software distribution in the folder *SDK/doc/idl* or on the Juniper Networks Web site at

http://www.juniper.net/techpubs/software/management/sdx/api-index.html

Verifying Idle Timeout Properties for the SAE

Review the following configuration properties to ensure that the settings are consistent with the requirements for your environment:

- Idle Timeout
- Adjust Session Time

To review idle timeout settings from SDX Configuration Editor:

1. In the navigation pane, expand SAE, and click a configuration object.
2. In the content pane, click the **Miscellaneous** tab.
3. Verify the setting for Idle Timeout(s).

   This value may be set in the service definition for the access service, or by the ISP in a RADIUS authorization response.

   An interval up to 5 minutes is typically recommended for the idle timeout. For the SRC software, the recommended minimum is 15 minutes.
4. In the Miscellaneous pane, expand Idle Timeout, and review the setting for Adjust Session Time. See the field description below.

Adjust Session Time

- When an idle timeout terminates a session, whether or not the session time reported in the accounting message is reduced by the idle time. This way the session time is accurately reported to avoid overcharges for the session.
- Value
  - True—Reduces the session time by the amount of time specified by Idle Timeout
  - False—Does not reduces the session time by the amount of time specified by Idle Timeout
- Default—True
- Property name—AccountingMgr.adjustSessionTime

Chapter 3

Configuring VoIP Services in an SRC Network

This chapter describes how the SRC network handles voice over IP (VoIP) services, and how to configure policies, services, and subscribers that support VoIP applications.

Topics in this chapter include:

- Overview of Session Management for VoIP Services on page 33
- Configuring Policies and Services for VoIP on page 34
- Activating VoIP Services for Assigned IP Subscribers on page 35
- Setting Timeouts for Assigned IP Subscriber Sessions on page 36

Overview of Session Management for VoIP Services

When the SAE activates a service session, it authorizes the session with authorization plug-ins; it may use the admission control plug-in (ACP) to perform call admission control and allocate bandwidth; and it installs the policy required for the service on a JUNOSe interface.

VoIP and multimedia service sessions are typically established in multiple phases that require changes to installed policies and authorized bandwidth while the service session remains active. To support VoIP sessions, the SAE allows changes to active service sessions. These changes include:

- Controlled bandwidth. If bandwidth demand increases, the authorization plug-in must authorize the change.
- Policy parameters. Only parameter substitution values can be changed. Policy parameters can include classifiers, such as destination address and port, and actions, such as rate-limit profiles.
- Session and idle timeouts. All attributes that can be set for initial service activation can be set for service session modifications.

Accounting and Tracking

Accounting information is preserved across service session changes. Accounting information for a complete service session includes the sum of counters for all service session segments.

When the ACP receives an interim update request, it compares the upstream and downstream bandwidth in the request with the current values. If the bandwidth has changed, ACP modifies its counters based on the difference between the current and new values.

Tracking plug-ins are informed of service session changes through an interim update message. The interim update is sent even if regular interim updates are disabled. If the controlled bandwidth changes, the interim update message contains the new bandwidth settings.

VoIP Call Setup

Initial setup of a VoIP call requires changes to bandwidth and to the endpoint address during call setup. The setup sequence for a VoIP call can follow this pattern:

1. The subscriber attempts to establish a call.
2. The gatekeeper (or Session Initiation Protocol [SIP] proxy) performs local admission control.
3. The gatekeeper allocates a Codec for the call; for example, 64 kbps.
4. The gatekeeper activates the VoIP service on the SAE with 64 kbps bandwidth and a destination address of unknown.
5. The SAE performs admission control, activates a service session, and installs policies on the router.
6. The gatekeeper negotiates call parameters with the remote endpoint.
7. The gatekeeper modifies the VoIP service with negotiated parameters; for example, 32 kbps, destination address 10.10.3.4, and UDP port 5678.
8. The SAE creates new policies that reflect changes to the traffic classifier and rate-limit profile, and then removes the existing policies from the router and installs the new policies.
9. The SAE sends interim updates to the ACP and tracking plug-ins.

Configuring Policies and Services for VoIP

When you set up a service that supports VoIP, you need to create a policy group for the VoIP service and assign the policy group to the VoIP service.

The SAE installs the policy on the router when the service is activated. When the service session is modified during VoIP call setup, the SAE replaces policy values with new values that were negotiated during call setup. The SAE then creates a new policy and installs it on the router.

When you set up a policy group for VoIP services, you need to assign variable parameters to fields that the SAE will need to modify. For example, source and destination addresses and UDP ports might be replaced with actual values. Upstream and downstream rate-limit parameters, such as committed rate and burst sizes, are likely to be modified.

Activating VoIP Services for Assigned IP Subscribers

When the SAE activates VoIP services, signaling proxies must identify subscriber equipment based on the IP address of the equipment. In the enterprise model, an IT manager typically subscribes to a service at a particular level in the subscriber hierarchy, and then provides the service to all access lines and subscribers who are at lower levels in the hierarchy. In cases such as this, the SAE manages the router interface but not the subscriber. The SAE does not know the IP addresses of the subscribers and therefore cannot provide the IP address to the signaling proxies.

A type of subscriber session called assigned IP supports the case in which the SAE does not manage the subscriber but needs to provide the IP address to signaling proxies. The SAE dynamically creates an assigned IP session based on an API call. The VoIP gateway must provide the following information to the SAE before the SAE can create the assigned IP session:

- The subscriber's IP address
- The name of a managed interface (The SAE applies policies for service sessions to this interface.)
- The name of the virtual router in which the managed interface resides

The NIC maps the subscriber's IP address to the SAE reference of the managing SAE, the interface name, and the virtual router name and provides this information to the VoIP gateway.

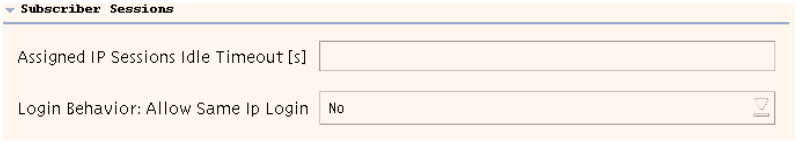
The network information collector (NIC) keeps track of managed interfaces through a NIC SAE plug-in agent. When an interface start, stop, or interim update event occurs, the SAE sends the interface tracking events to the NIC SAE plug-in agent. The NIC uses this information as part of the process of creating these mappings.

For more information, see *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 11, Configuring NIC on a Solaris Platform*.

Setting Timeouts for Assigned IP Subscriber Sessions

To set timeouts for assigned IP subscriber sessions in the SAE configuration:

- In SDX Configuration Editor, configure the Assigned IP Sessions Idle Timeout field in the Miscellaneous tab.



Assigned IP Sessions Idle Timeout [s]

- Interval after which assigned IP subscriber sessions are deactivated if no service session is active.
- Value—Number of seconds in the range 0–2147483647
- Default—900
- Property name—UserManager.assignedIp.idletimeout

Chapter 4

Providing Premium Services in a PCMM Environment

This chapter describes the SRC application's support for the *PacketCable Multimedia Specification* (PCMM) as defined by Cable Television Laboratories, Inc. (CableLabs).

Topics in this chapter include:

- Overview of a PCMM Environment on page 37
- Using the SAE in a PCMM Environment on page 47

Overview of a PCMM Environment

The PCMM specification defines a standards-based way to deliver premium quality of service (QoS)-enhanced services across the radio frequency (RF) portion of a cable network. The PCMM capabilities of the SRC software along with Juniper Networks routers provide an end-to-end solution that seamlessly links the cable operator's RF domain with IP edge and core QoS services.

Key services supported in this environment include:

- Bandwidth on demand and variable bandwidth
- QoS-enabled streaming media, including video on demand and video telephony
- Residential voice over IP (VoIP)
- Multicast audio and video applications
- Videoconferencing
- Interactive gaming
- Peer-to-peer controls and protection services

References

For more information about PCMM, consult the following specifications provided by CableLabs:

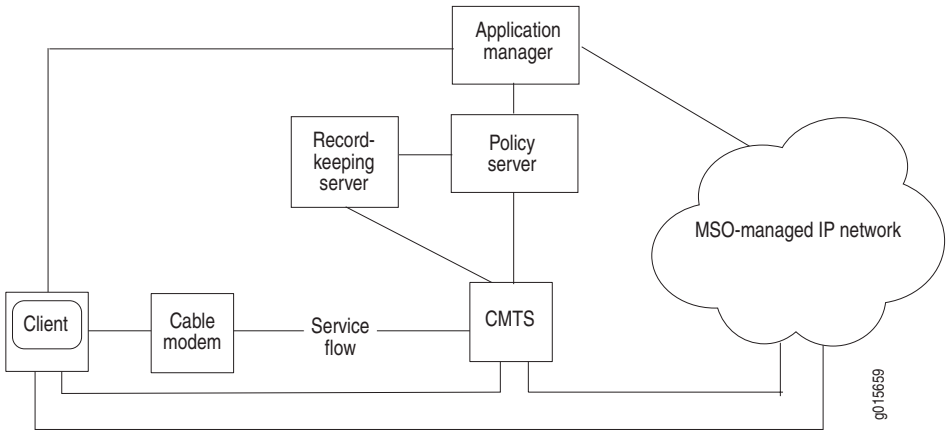
- PacketCable Multimedia Architecture Framework Technical Report (PKT-TR-MM-ARCH)
- PacketCable Multimedia Specification PKT-SP-MM-I03-051221
- PacketCable Security Specifications (PKT-SP-SEC)

PCMM Architecture

Figure 6 depicts the PCMM architectural framework. The basic roles of the various PCMM components are:

- Application manager—Provides an interface to policy server(s) for the purpose of requesting QoS-based service on behalf of a subscriber or a network management system. It maps session requests to resource requests and creates policies.
- Policy server—Acts as a policy decision point and policy enforcement point and manages relationships between application managers and cable modem termination system (CMTS) devices.
- CMTS device—Cable modem termination system. Performs admission control and manages network resources through Data over Cable Service Interface Specifications (DOCSIS) service flows.
- Client—Represents endpoints, such as PC applications, that can send or receive data.
- Record-keeping server—Receives event messages from other network elements, such as the policy server or CMTS device, and acts as a short-term repository for the messages. It can also assemble event messages into coherent sets or call detail records, which are then made available to other back office systems, such as billing, fraud detection, and other systems.

Figure 6: PCMM Architectural Framework



In the PCMM architecture, a client requests a multimedia service from an application manager. The application manager relays the request to a policy server. The policy server is then responsible for provisioning the policies on a CMTS device. Based on the request, the policy server records an event that indicates the policy request. The request can include network resource records, and the policy server can provide the records to a record-keeping server, such as a RADIUS accounting server.

The policy server may also provide functions such as tracking resource usage and tracking the authorization of resources on a per-subscriber, per-service, or aggregate basis.

DOCSIS Protocol

The DOCSIS protocol is the standard for providing quality of service for traffic between the cable modem and CMTS devices. The CMTS device is the headend in the DOCSIS architecture, and it controls the operations of many cable modems. Two channels carry signals between CMTS devices and cable modems:

- Downstream channels—Carry signals from the CMTS headend to cable modems.
- Upstream channels—Carry signals from the cable modems to the CMTS headend.

The DOCSIS protocol defines the physical layer and the Media Access Control (MAC) protocol layer that is used on these channels.

A cable modem usually uses one upstream channel and an associated downstream channel. Upstream channels are shared, and the CMTS device uses the MAC protocol to control the cable modem's access to the upstream channel.

Service Flows

The DOCSIS protocol uses the concept of service flows to support QoS on upstream and downstream channels. A service flow is a unidirectional flow of packets that provides a particular quality of service. Traffic is classified into a service flow, and each service flow has its own set of QoS parameters. The SRC software is compliant with the following upstream service flow scheduling types, as defined in the PacketCable Multimedia Specification PKT-SP-MM-I03-051221.

- Best effort—Used for standard Internet traffic such as Web browsing, e-mail, or instant messaging.
- Non-real-time polling service (NRTPS)—Used for standard Internet traffic that requires high throughput, and traffic that requires variable-sized data packets on a regular basis, such as high-bandwidth File Transfer Protocol (FTP).
- Real-time polling service (RTPS)—Used for applications such as Moving Pictures Experts Group (MPEG) video.

- Unsolicited grant service (UGS)—Used for real-time traffic that generates fixed-size data packets on a periodic basis. Applications include VoIP.

- Unsolicited grant service with activity detection (UGS-AD)—Used for applications such as voice activity detection, also known as silence suppression.

Downstream service flows are defined through a similar set of QoS parameters that are associated with the best-effort scheduling type on upstream service flows.

See *Delivering QoS Services in a Cable Environment* in *SDX Services and Policies Guide, Chapter 6, Policy Management Overview* for more information about each scheduling type.

Client Types

The PCMM specification uses the concept of clients and defines a client as a logical entity that can send or receive data. The SRC software supports type 1 and type 2 clients.

The PCMM specification defines two resource reservation models for each client type—a single phase and a dual phase. The SRC software supports the single-phase model.

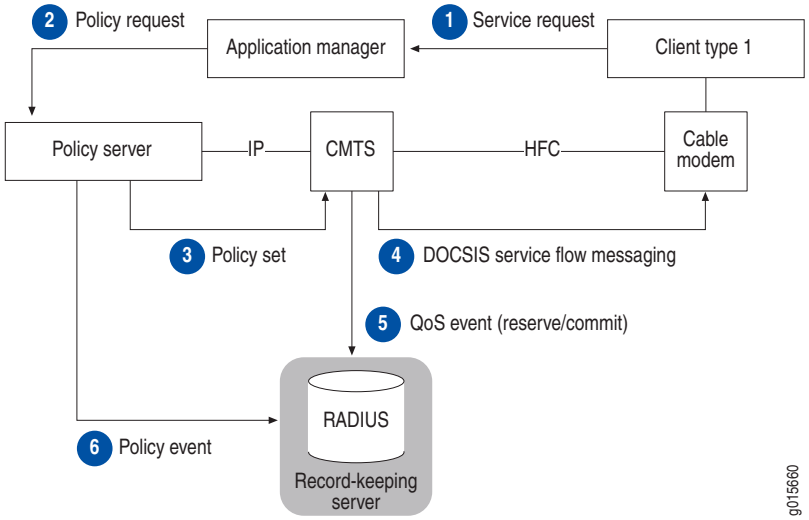
Client Type 1 Single Phase Resource Reservation Model

Type 1 clients represent endpoints, such as PC applications or gaming consoles, that lack specific QoS awareness or signaling capabilities. Type 1 clients communicate with an application manager to request a service. They do not request QoS resources directly from the multiple service operator (MSO) network.

Client type 1 entities support the proxied-QoS with policy-push scenario of service delivery defined in PacketCable Multimedia Architecture Framework Technical Report (PKT-TR-MM-ARCH). In this scenario, the application manager requests QoS resources on behalf of the client, and the policy server pushes the request to the CMTS device. The CMTS device sets up and manages the DOCSIS service flow that the application requires, and might also set up and manage the cable modems.

Figure 7 shows the message flow in an application scenario for the client type 1 single-phase resource reservation model.

Figure 7: Client Type 1 Single-Phase Resource Reservation Model



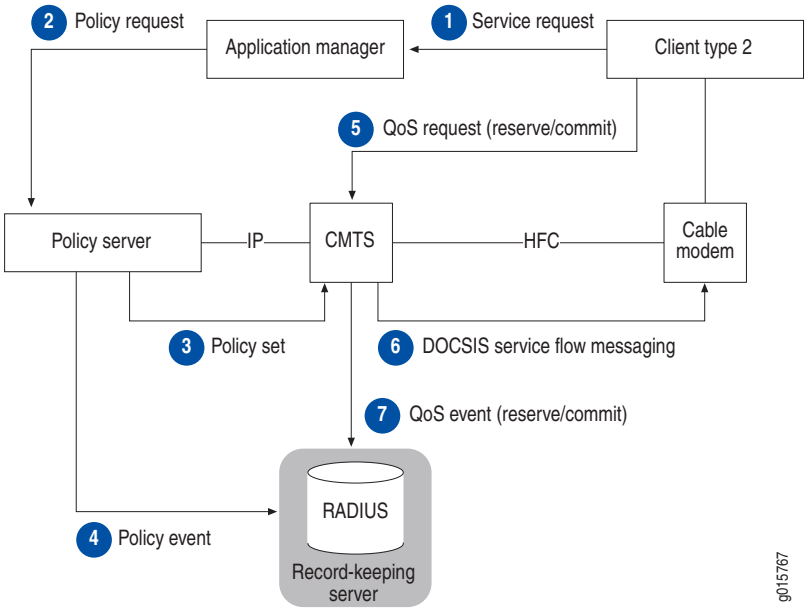
Client Type 2 Single Phase Resource Reservation Model

Type 2 clients represent endpoints that have QoS awareness or signaling capabilities. Type 2 clients communicate with an application manager to request a service and to obtain a token to present for requesting QoS resources directly from the MSO network.

Client type 2 entities support the client-requested QoS with policy-push scenario of service delivery defined in PacketCable Multimedia Architecture Framework Technical Report (PKT-TR-MM-ARCH). In this scenario, the application manager requests QoS resources on behalf of the client, and the policy server pushes the request to the CMTS device. The CMTS device sets up and manages the DOCSIS service flow that the application requires. After the CMTS device sets up the policy, the client can request QoS resources directly from the CMTS device as long as the request is authorized by the policy server.

Figure 8 shows the message flow in an application scenario for the client type 2 single-phase resource reservation model.

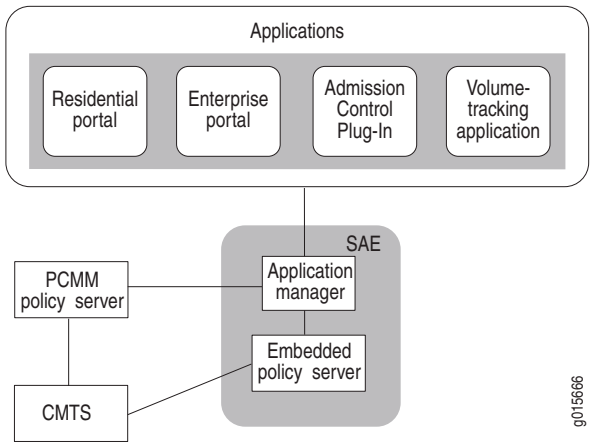
Figure 8: Client Type 2 Single-Phase Resource Reservation Model



SRC Software in the PCMM Environment

Figure 9 shows the SRC software in the PCMM environment. The SAE is an application manager that can manage a PCMM-compliant policy server and/or a CMTS device on behalf of applications. The SAE has an embedded policy server that is not fully PCMM-compliant, but it can manage CMTS devices without requiring an external policy server. The Juniper Policy Server (JPS), a component of the SRC software that acts as a policy server, is a PCMM-compliant policy server. For more information about using the JPS, see *Chapter 11, Using PCMM Policy Servers*.

Figure 9: SRC Software in the PCMM Environment



Traffic Profiles

The SRC software supports three types of policies that you can use to define traffic profiles between the CMTS device and the cable modem:

- DOCSIS parameters—Specifies the traffic profile through DOCSIS-specific parameters. You select the type of service flow that you want to offer, and then configure QoS parameters for the service flow.
- Service class name—Specifies the name of a service class that is configured on the CMTS device.
- FlowSpec—Defines the traffic profile through an Resource Reservation Protocol (RSVP)-like parameterization scheme. FlowSpecs support both controlled-load and guaranteed services.

You can also mark packets and then install policies that handle the marked packets in a certain way. The mark action sets the ToS byte in the IP header of IPv4 traffic or the traffic-class field in the IP header of IPv6 traffic.

See *Delivering QoS Services in a Cable Environment* in *SDX Services and Policies Guide, Chapter 6, Policy Management Overview* for more information about traffic profiles.

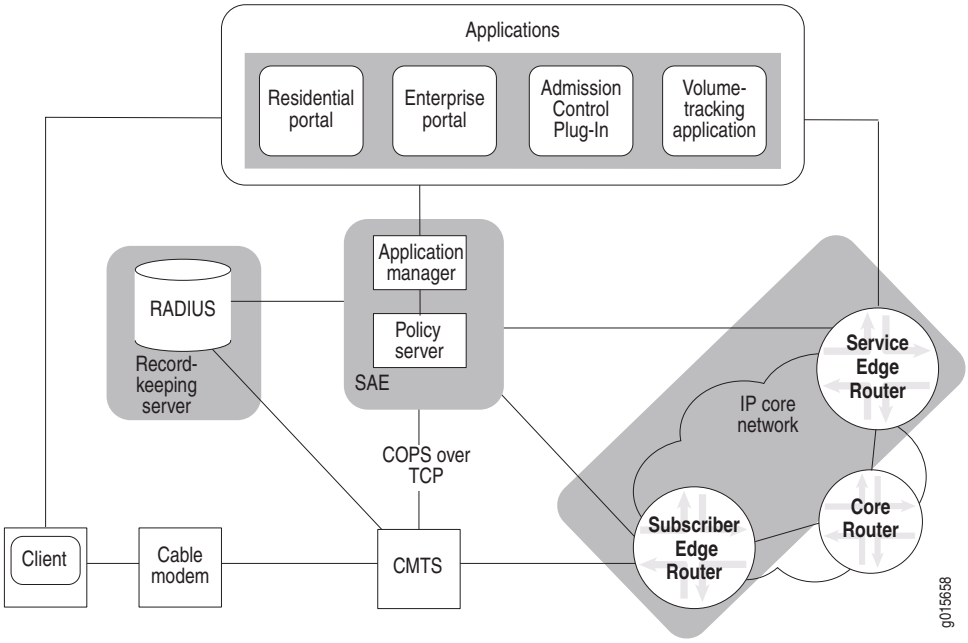
End-to-End QoS Architecture

The previous sections show how the SRC software supports QoS in the cable operator's RF domain, which encompasses the connection from the cable modem to the CMTS device. Using the SRC software along with Juniper Networks routers, you can link the RF domain to the subscriber and service edge domains.

- IP subscriber edge domain—Includes the IP network from the CMTS device to the edge router that typically connects to the cable operator's regional access network. (See *Extending QoS to the Subscriber Edge Domain* on page 44.)
- IP service edge domain—Typically includes the IP network that connects the data center that houses service delivery applications to a backbone or directly to a cable head-on facility. (See *Extending QoS to the Service Edge Domain* on page 44.)

By provisioning services across a network path, you can deliver a particular level of service for specified types of traffic. Figure 10 on page 44 shows a typical high-level architecture of a cable operator and how the SRC software and Juniper Networks routers can be deployed to deliver end-to-end QoS services.

Figure 10: End-to-End QoS Architecture in a Cable Network



Extending QoS to the Subscriber Edge Domain

The subscriber edge domain includes subscriber edge routers that aggregate CMTS devices. To support QoS in subscriber edge domains, QoS must be enabled across the subscriber edge into the core or regional access network. When the SRC software receives a service request, it performs service authorization, which can include admission control. It then sends policies to the appropriate CMTS device and subscriber edge router interface.

In addition to the QoS services required in the RF domain, service policies in the subscriber edge domain that must be available for provisioning at this point include:

- Policy routing to best-of-breed appliances and premium paths
- Rate limiting, traffic shaping, and marking
- Admission control (edge resources and core resources)
- Captive portal and Web redirect capabilities
- Filtering and JUNOS routing platform-based firewall services
- JUNOS routing platform virtual private network (VPN) services

Extending QoS to the Service Edge Domain

The service edge domain includes service edge routers that aggregate applications. To support QoS in service edge domains, the SRC software sends policies to a service edge router that provides for enhanced service delivery to the service origination edge for centralized or hosted services, such as multimedia or VoD.

In addition to the QoS services required in the RF domain, service policies in the service edge domain that must be capable of being provisioned at this point include:

- Policy routing to best-of-breed appliances and premium paths
- Rate limiting, traffic shaping (called hierarchical queuing in JUNOSe software), and marking
- Filtering and JUNOS routing platform based firewall services
- JUNOS routing platform VPN services

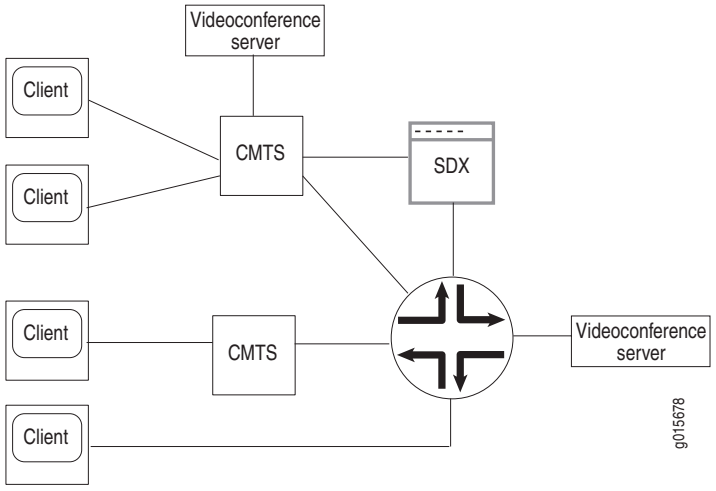
Provisioning End-to-End Services

The following sections provide examples of how you can use the SRC software to provision services for video applications. Although the examples show one SAE managing all the network devices, separate SAEs could manage each device and provide the same service.

Example for Videoconferencing Services

You can configure services to mark traffic forwarded from specified systems, and then apply an end-to-end service level for that traffic. Figure 11 shows a scenario in which videoconferencing is delivered in a PCMM environment.

Figure 11: Videoconferencing Example



To ensure a specified level of service from each client PC to the videoconference server and then to each client PC participating in the videoconference, you could configure the following types of services:

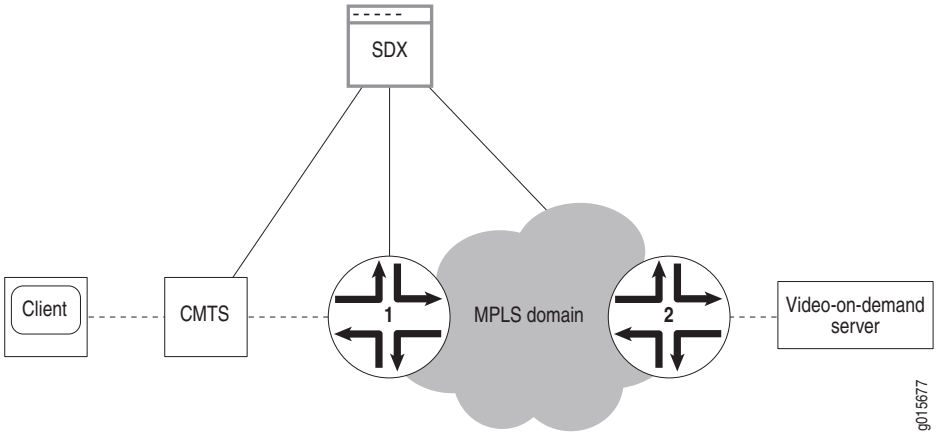
- Three services:
  - A service that provides policies to mark packets with a specified type of service for the videoconferencing software.
  - A service that provides policies for the type of service specified for CMTS device.
  - A service that provides policies for the type of service specified for the JUNOS routing platform or JUNOSe router.
- An infrastructure service for each service.
- An aggregate service that contains the three infrastructure services as fragment services.

This configuration marks packets that the CMTS device receives from both client and server, and applies forwarding policies on the CMTS device and on the JUNOSe router or JUNOS routing platform for packets sent to and received from the videoconferencing server.

Example for Video-on-Demand Services

You can configure services to provide server-to-client service for traffic sent from a video-on-demand server to client PCs. Figure 12 shows a scenario in which video on demand is delivered in a PCMM environment.

Figure 12: Video-on-Demand Example



To ensure a specified level of service from the video-on-demand server to the client PC, you could configure the following types of services:

- Services that provide bandwidth-on-demand (BoD) policies for traffic that is being forwarded from the video-on-demand server through:
  - JUNOS routing platforms
  - CMTS devices
- A script service that sets up the Multiprotocol Label Switching (MPLS) path and delivers the specified service level for traffic that is being forwarded from the video-on-demand server through the MPLS domain.
- An infrastructure service for each value-added and script service.
- An aggregate service that contains all the infrastructure services as fragment services.

This configuration applies BoD policies to the two JUNOS routing platforms, the MPLS domain, and the CMTS device, and sets up the MPLS path from JUNOS routing platform (2) to JUNOS routing platform (1).

Using the SAE in a PCMM Environment

The SAE uses the Common Open Policy Service (COPS) protocol as specified in the PacketCable Multimedia Specification PKT-SP-MM-I03-051221 to manage PCMM-compliant CMTS devices in a cable network environment. The SAE connects to the CMTS device by using a COPS over Transmission Control Protocol (TCP) connection. In cable environments, the SAE manages the connection to the CMTS device.

The CMTS device does not provide address requests or notify the SAE of new subscribers, subscriber IP addresses, or any other attributes. IP address detection and all other subscriber attributes are collected outside of the COPS connection to the CMTS device. The SAE uses COPS only to push policies to the CMTS device and to learn about the CMTS status and usage data.

Because the CMTS device does not have the concept of interfaces, the SRC software uses pseudointerfaces to model CMTS subscriber connections similar to subscriber connections for JUNOS routing platforms and JUNOSe routers.

This section describes how the SAE is used in cable networks. It includes the following topics:

- Logging In Subscribers and Creating Sessions on page 48
- SAE Communities on page 51
- Storing Session Data on page 52

Logging In Subscribers and Creating Sessions

You can use two mechanisms to obtain subscriber address requests and other information and to set up a pseudointerface on the CMTS device. (You must choose one mechanism; you cannot mix them.):

1. Assigned IP subscriber. The SAE learns about a subscriber through subscriber-initiated activities, such as activating a service through the portal or through the Advanced Services Gateway (ASG).

   With this method, you use the assigned IP subscriber login type along with the network interface collector (NIC) to map IP addresses to the SAE.

2. Event notification from an IP address manager. The SAE learns about subscribers through notifications from an external IP address manager, such as a DHCP server or a RADIUS server.

   With this method, you use the event notification application programming interface (API). The API provides an interface to the IP address manager, and lets the IP address manager notify the SAE of events such as IP address assignments.

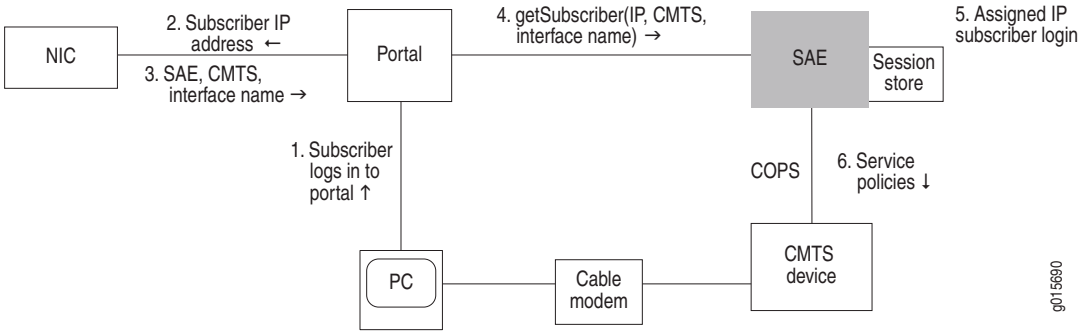
Assigned IP Subscribers

With the assigned IP subscriber method of logging in subscribers and creating sessions, the SRC software uses IP address pools along with NIC resolvers to provide mapping of IP addresses to SAEs. You configure the static address pools or dynamically discovered address pools in the virtual router configuration for a CMTS device. These pools are published in the NIC. The NIC maps subscriber IP addresses in requests received through the portal or Advanced Services Gateway to the SAE that currently manages that CMTS device.

Login Interactions with Assigned IP Subscribers

This section describes login interactions for assigned IP subscribers. In the example shown in Figure 13, the subscriber activates a service through a portal. You could also have the subscriber activate a service through the Advanced Services Gateway.

Figure 13: Login Interactions with Assigned IP Subscribers



The sequence of events for logging in and creating sessions for assigned IP subscribers is:

1. The subscriber logs in to the portal.
2. The portal sends the subscriber's IP address to the NIC.
3. Based on the IP address, the NIC looks up the subscriber's SAE, CMTS device, and interface name, and returns this information to the portal.
4. The portal sends a getSubscriber message to the SAE. The message includes the subscriber's IP address, CMTS device, and interface name.
5. The SAE creates an assigned IP subscriber and performs a subscriber login. Specifically, it:
    a. Runs the interface classification script and creates a pseudointerface for the PCMM device driver.
        - If it finds a default policy, it pushes the policy to the CMTS device.
        - If it does not find a default policy, it continues with the next steps.
    b. Runs the subscriber classification script with the IP address of the subscriber. (Use the ASSIGNEDIP login type in subscriber classification scripts.)
    c. Loads the subscriber profile.
    d. Runs the subscriber authorization plug-ins.
    e. Runs the subscriber tracking plug-ins.
    f. Creates a subscriber session and stores the session data in the session store file.
6. The SAE pushes service policies for the subscriber session to the CMTS device.

Because the SAE is not notified when the subscriber logs out, the assigned IP idle timer begins when no service is active. The SAE removes the interface subscriber session when the timeout period ends.

Event Notification from an IP Address Manager

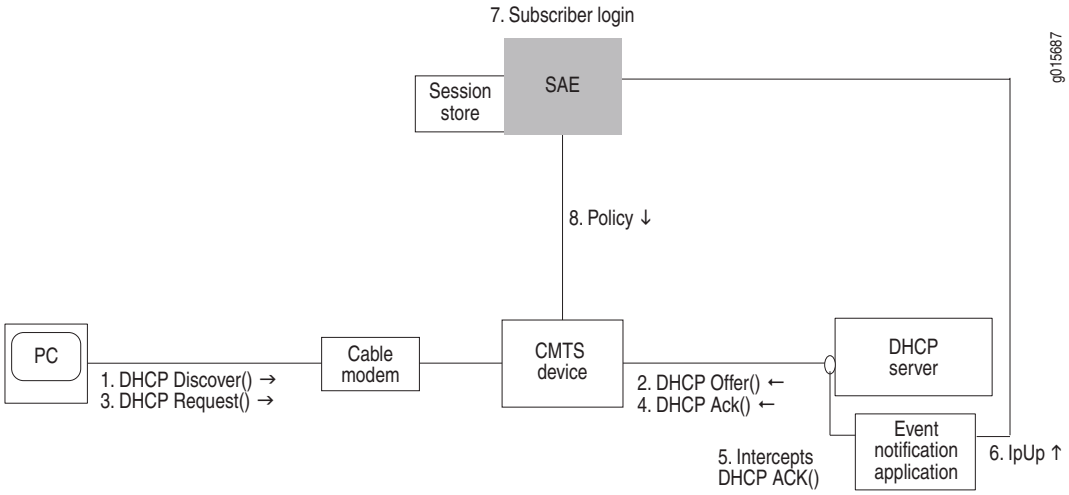
With the event notification method of logging in subscribers and creating subscriber sessions, the subscriber logs in to the CMTS device and obtains an IP address through an address server, usually a DHCP server. The SAE receives notifications about the subscriber, such as the subscriber's IP address, from an event notification application that is installed on the DHCP server.

To use this method of logging in subscribers, you can use the event notification API to create the application that notifies the SAE when events occur between the DHCP server and the CMTS device. You can also use Monitoring Agent, an application that was created with the event notification API, and that monitors DHCP or RADIUS messages for DHCP or RADIUS servers. See *SRC Application Library Guide, Chapter 27, Integrating IP Address Managers with the SAE*.

Login with Event Notification

This section describes login interactions using event notifications.

Figure 14: Login Interactions with Event Notification Application



The sequence of events for logging in subscribers and creating sessions is:

1. The DHCP client in the subscriber's computer sends a DHCP discover request to the DHCP server.
2. The DHCP server sends a DHCP offer to the subscriber's DHCP client.
3. The DHCP client sends a DHCP request to the DHCP server.
4. The DHCP server acknowledges the request by sending a DHCP Ack message to the DHCP client.
5. The event notification application that is running on the DHCP server intercepts the DHCP Ack message.
6. The event notification application sends an ipUp message to the SAE that notifies the SAE that an IP address is up.

7. The SAE performs a subscriber login. Specifically, it:

    a. Runs the interface classification script and creates a pseudointerface for the PCMM device driver.

        - If it finds a default policy, it pushes the policy to the CMTS device.
        - If it does not find a default policy, it continues with the next steps.

    b. Runs the subscriber classification script.

    c. Loads the subscriber profile.

    d. Runs the subscriber authorization plug-ins.

    e. Runs the subscriber tracking plug-ins.

    f. Creates a subscriber session and stores the session in the session store file.

8. The SAE provisions policies for the subscriber session on the CMTS device.

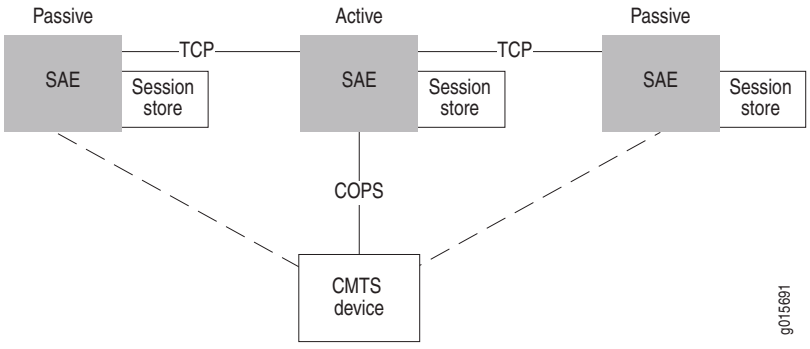
The ipUp event should be sent with a timeout set to the DHCP lease time. The DHCP server sends an ipUp event for each Ack sent to the client. The SAE restarts the timeout each time it receives an ipUp event.

If the client explicitly releases the DHCP address (that is, it sends a DHCP release event), the DHCP server sends an ipDown event. If the client does not renew the address, the lease expires on the DHCP server and the timeout expires on the SAE.

SAE Communities

For SAE redundancy in a cable network, you can have a community of two or more SAEs. SAEs in a community are given the role of either active SAE or passive SAE. The active SAE manages the connection to the CMTS device and keeps session data up to date within the community. Figure 15 shows a typical SAE community.

Figure 15: SAE Community



When an SAE opens a connection to the CMTS device, it negotiates with other SAEs to determine which SAE controls the CMTS device. The SAE community manager and members of the community select the active SAE.

A passive SAE needs to take over as active SAE in any of the following cases:

- The active SAE shuts down or the connection between the CMTS device and the active SAE goes down. In this case, the active SAE notifies the passive SAEs, and one of the passive SAEs takes over as active SAE.
- A passive SAE does not receive a keepalive message from the active SAE within the keepalive interval. In this case, the passive SAE attempts to become the active SAE.

Storing Session Data

To aid in recovering from an SAE failover, the SAE stores subscriber and service session data. When the SAE manages a CMTS device, session data is stored locally in the SAE host's file system. The SRC component that controls the storage of session data on the SAE is called the session store. The session store queues data and then writes the data to session store files on the SAE host's disk. Once the data is written to disk, it can survive a server reboot.

For more information, see *Storing Subscriber and Service Session Data* in *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 3, Configuring the SAE with SDX Configuration Editor*.

PCMM Record-Keeping Server Plug-In

To allow the SAE's embedded policy server to communicate with a record-keeping server (RKS) in a PCMM environment, you need to use the PCMM record-keeping server plug-in. This plug-in is similar to the RADIUS accounting plug-ins, but it works with any RKS that is compliant with the PCMM specification. The RKS plug-in supports additional attributes: Application-Manager-ID, Request-Type, and Update-Reason. The plug-in sends all requests to the RKS as Acct-Status-Type = Interim-Update.

Chapter 5

Configuring the SAE for a PCMM Environment with the SRC CLI

This chapter shows how to set up the SAE for a PacketCable Multimedia Specification (PCMM) environment with the SRC CLI. You can also use SDX Configuration Editor to configure the SAE on a Solaris platform. See *Chapter 6, Configuring the SAE for a PCMM Environment with SDX Configuration Editor*.

Topics in this chapter include:

- Overview of Configuring the SAE for a Cable Network Environment on page 53
- Configuring the SAE to Manage PCMM Devices on page 54
- Setting Up SAE Communities on page 57
- Configuring SAE Properties for the Event Notification API on page 58
- Configuring Record-Keeping Server Peers for Plug-Ins on page 59
- Configuring PCMM Record-Keeping Server Plug-Ins on page 60
- Configuring CMTS-Specific RKS Plug-Ins on page 63

Overview of Configuring the SAE for a Cable Network Environment

The tasks to configure the SAE for a cable network environment are:

1. Configuring the SAE to Manage PCMM Devices on page 54.
2. Configuring the Session Store Feature.

   See *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 2, Configuring the SAE with the SRC CLI*.
3. Setting Up SAE Communities on page 57.
4. (Optional) Configuring SAE Properties for the Event Notification API on page 58 (if you are using an external address manager).

5. (Optional) Configuring Record-Keeping Server Peers for Plug-Ins on page 59 (if you are using the RKS plug-in).

6. (Optional) Configuring PCMM Record-Keeping Server Plug-Ins on page 60 (if you are using the SAE's embedded policy server).

7. (Optional) Configuring CMTS-Specific RKS Plug-Ins on page 63.

In addition to configuring the SAE, you need to:

1. Configure the CMTS device in the directory (if you are using the SAE's embedded policy server).

   See *Adding Objects for CMTS Devices with the SRC CLI* on page 77.

2. Configure the NIC (if you are using assigned IP subscribers).

   See *Chapter 9, Using the NIC Resolver in a PCMM Environment*.

3. Enable the Common Open Policy Service (COPS) interface on the CMTS device. See the documentation for your CMTS device for information about how to do this.

Configuring the SAE to Manage PCMM Devices

The SAE connects to the PCMM device by using a COPS over TCP connection. The PCMM device driver controls this connection.

Use the following configuration statements to configure the SAE to manage CMTS devices:

```
shared sae configuration driver pcmm {
    keepalive-interval keepalive-interval;
    tcp-connection-timeout tcp-connection-timeout;
    application-manager-id application-manager-id;
    message-timeout message-timeout;
    cops-message-maximum-length cops-message-maximum-length;
    cops-message-read-buffer-size cops-message-read-buffer-size;
    cops-message-write-buffer-size cops-message-write-buffer-size;
    sae-community-manager sae-community-manager;
    disable-full-sync disable-full-sync;
    disable-pcmm-i03-policy disable-pcmm-i03-policy;
    session-recovery-retry-interval session-recovery-retry-interval;
    element-id element-id;
    default-rks-plug-in default-rks-plug-in;
}
```

To configure the SAE to manage CMTS devices:

1. From configuration mode, access the configuration statement that configures the PCMM driver. In this sample procedure, the PCMM device driver is configured in the west-region group.

   user@host# **edit shared sae group west-region configuration driver pcmm**

2. Configure the interval between keepalive messages sent from the COPS client (the PCMM device) to the COPS server (the SAE).

   [edit shared sae group west-region configuration driver pcmm]
   user@host# **set keepalive-interval** *keepalive-interval*

3. Configure the timeout for opening a TCP connection to the PCMM device.

   [edit shared sae group west-region configuration driver pcmm]
   user@host# **set tcp-connection-timeout** *tcp-connection-timeout*

4. When this SAE is configured as the application manager, configure the identifier of the application manager.

   [edit shared sae group west-region configuration driver pcmm]
   user@host# **set application-manager-id** *application-manager-id*

5. Configure the time that the COPS server (the SAE) waits for a response to COPS requests from the COPS client (the PCMM device). Change this value only if a high number of COPS timeout events appear in the error log.

   [edit shared sae group west-region configuration driver pcmm]
   user@host# **set message-timeout** *message-timeout*

6. Configure the maximum length of a COPS message.

   [edit shared sae group west-region configuration driver pcmm]
   user@host# **set cops-message-maximum-length** *cops-message-maximum-length*

7. Configure the buffer size for receiving COPS messages from the COPS client.

   [edit shared sae group west-region configuration driver pcmm]
   user@host# **set cops-message-read-buffer-size** *cops-message-read-buffer-size*

8. Configure the buffer size for sending COPS messages to the COPS client.

   [edit shared sae group west-region configuration driver pcmm]
   user@host# **set cops-message-write-buffer-size** *cops-message-write-buffer-size*

9. Configure the name of the community manager that manages PCMM driver communities. Active SAEs are selected from this community.

   [edit shared sae group west-region configuration driver pcmm]
   user@host# **set sae-community-manager** *sae-community-manager*

10. Enable or disable state synchronization with PCMM policy servers.

    [edit shared sae group west-region configuration driver pcmm]
    user@host# **set disable-full-sync** *disable-full-sync*

11. Enable or disable the SAE to send classifiers to the router that comply with PCMM IO3. Disable this option if your network deployment has CMTS devices that do not support PCMM IO3.

    [edit shared sae group west-region configuration driver pcmm]
    user@host# **set disable-pcmm-i03-policy** *disable-pcmm-i03-policy*

12. Configure the time between attempts by the SAE to restore service sessions that are being recovered in the background when state synchronization completes with a state-data-incomplete error.

    [edit shared sae group west-region configuration driver pcmm]
    user@host# **set session-recovery-retry-interval** *session-recovery-retry-interval*

13. (Optional) Configure the unique identifier that the SAE uses to identify itself when it originates in record-keeping server (RKS) events.

    [edit shared sae group west-region configuration driver pcmm]
    user@host# **set element-id** *element-id*

14. (Optional) Specify the name of the default RKS plug-in to which the SAE sends events for CMTS devices.

    [edit shared sae group west-region configuration driver pcmm]
    user@host# **set default-rks-plug-in** *default-rks-plug-in*

15. (Optional) Verify your PCMM driver configuration.

```
[edit shared sae group west-region configuration driver pcmm]
user@host# show
keepalive-interval 45;
tcp-connection-timeout 5;
application-manager-id 1;
message-timeout 120000;
cops-message-maximum-length 204800;
cops-message-read-buffer-size 3000;
cops-message-write-buffer-size 3000;
sae-community-manager PcmmCommunityManager;
disable-full-sync true;
disable-pcmm-i03-policy true;
session-recovery-retry-interval 3600000;
element-id 1;
default-rks-plug-in rksTracking;
```

Related Information

For additional information, see the following sources:

- For information about setting up SAE groups, see *SDX Getting Started Guide, Chapter 16, Setting Up an SAE with the SRC CLI*.
- For information about setting up a community manager, see See *Setting Up SAE Communities* on page 57.

Setting Up SAE Communities

You can configure the following for SAE communities:

- Define the members of an SAE community by adding the IP addresses of SAEs in the community to the virtual router object of the network device in the directory.

  See *Creating Virtual Routers for the CMTS Device with the SRC CLI* on page 78.

- Configure parameters for the SAE community manager.

  See *Configuring the SAE Community Manager* on page 57.

- Specify the name of the community manager with the **set sae-community-manager** option in the PCMM driver configuration.

  See *Configuring the SAE to Manage PCMM Devices* on page 54.

- If there is a firewall in the network, configure the firewall to allow SAE messages through.

Configuring the SAE Community Manager

Use the following configuration statements to configure the SAE community manager that manages PCMM device communities:

```
shared sae configuration external-interface-features name CommunityManager {
    keepalive-interval keepalive-interval;
    threads threads;
    acquire-timeout acquire-timeout;
    blackout-time blackout-time;
}
```

To configure the community manager:

1. From configuration mode, access the configuration statements for the community manager. In this sample procedure, west_region is the name of the SAE group, and sae_mgr is the name of the community manager.

   ```
   user@host# edit shared sae group west-region configuration
   external-interface-features sae_mgr CommunityManager
   ```

2. Specify the interval between keepalive messages sent from the active SAE to the passive members of the community.

   ```
   [edit shared sae group west-region configuration external-interface-features
   sae_mgr CommunityManager]
   user@host# set keepalive-interval keepalive-interval
   ```

3. Specify the number of threads that are allocated to manage the community. You generally do not need to change this value.

   ```
   [edit shared sae group west-region configuration external-interface-features
   sae_mgr CommunityManager]
   user@host# set threads threads
   ```

4. Specify the amount of time an SAE waits for a remote member of the community when it is acquiring a distributed lock. You generally do not need to change this value.

   [edit shared sae group west-region configuration external-interface-features sae_mgr CommunityManager]
   user@host# **set acquire-timeout** *acquire-timeout*

5. Specify the amount of time that an active SAE must wait after it shuts down before it can try to become the active SAE of the community again.

   [edit shared sae group west-region configuration external-interface-features sae_mgr CommunityManager]
   user@host# **set blackout-time** *blackout-time*

6. (Optional) Verify the configuration of the SAE community manager.

```
[edit shared sae group west-region configuration external-interface-features
sae_mgr CommunityManager]
user@host# show
CommunityManager {
  keepalive-interval 30;
  threads 5;
  acquire-timeout 15;
  blackout-time 30;
}
```

Related Information

For additional information, see the following source:

- For information about setting up SAE groups, see *SDX Getting Started Guide, Chapter 16, Setting Up an SAE with the SRC CLI*.

Configuring SAE Properties for the Event Notification API

Use the following configuration statements to configure properties for the Event Notification API:

shared sae configuration external-interface-features *name* EventAPI {
    retry-time *retry-time*;
    retry-limit *retry-limit*;
    threads *threads*;
}

To configure properties for the Event Notification API:

1. From configuration mode, access the configuration statements for the Event Notification API. In this sample procedure, west-region is the name of the SAE group, and event_api is the name of the Event API configuration.

   user@host# **edit shared sae group west-region configuration external-interface-features event_api EventAPI**

2. Specify the amount of time between attempts to send events that could not be delivered.

   [edit shared sae group west-region configuration external-interface-features event_api EventAPI]
   user@host# **set retry-time** *retry-time*

3. Specify the number of times an event fails to be delivered before the event is discarded.

   [edit shared sae group west-region configuration external-interface-features event_api EventAPII]
   user@host# **set retry-limit** *retry-limit*

4. Specify the number of threads allocated to process events.

   [edit shared sae group west-region configuration external-interface-features event_api EventAPI]
   user@host# **set threads** *threads*

5. (Optional) Verify the configuration of the Event Notification API properties.

```
[edit shared sae group west-region configuration external-interface-features
event_api EventAPI]
user@host# show
EventAPI {
  retry-time 300;
  retry-limit 5;
  threads 5;
}
```

Related Information

For additional information, see the following source:

- For information about setting up SAE groups, see *SDX Getting Started Guide, Chapter 16, Setting Up an SAE with the SRC CLI*.

Configuring Record-Keeping Server Peers for Plug-Ins

An RKS peer is an instance of an RKS. A PCMM environment has a primary RKS and optionally a secondary RKS. The primary RKS is mandatory, and you assign the RKS as primary by configuring it as the default peer in the RKS plug-in. The secondary RKS is optional, and it is an RKS peer that is not configured as the default peer. If you define multiple nondefault peers, one of them is randomly chosen to be the secondary RKS.

RKS peers are configured in the peer group for each PCMM RKS plug-in instance. To create an RKS peer group:

Use the following configuration statements to configure an RKS peer group.

shared sae configuration plug-ins pool *name* pcmm-rks peer-group *name* {
    server-address *server-address*;
    server-port *server-port*;
}

To configure an RKS peer group:

1. From configuration mode, access the configuration statements for RKS plug-ins. In this sample procedure, west-region is the name of the SAE group, and rksPlugin is the name of the plug-in and rksPeer is the name of the peer group.

   user@host# **edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks peer-group rksPeer**

2. Specify the IP address of the RKS server to which the SAE sends accounting data.

   [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks peer-group rksPeer]
   user@host# **set server-address** *server-address*

3. Specify the port used for sending accounting packets.

   [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks peer-group rksPeer]
   user@host# **set server-port** *server-port*

4. (Optional) Verify your configuration.

```
[edit shared sae group west-region configuration plug-ins pool rksPlugin
pcmm-rks peer-group rksPeer]
user@host# show
server-address 10.10.3.60;
server-port 1812;
```

Related Information

For additional information, see the following source:

- For information about setting up SAE groups, see *SDX Getting Started Guide, Chapter 16, Setting Up an SAE with the SRC CLI*.

Configuring PCMM Record-Keeping Server Plug-Ins

Use the following configuration statements to configure an RKS plug-in.

```
shared sae configuration plug-ins pool name pcmm-rks {
    load-balancing-mode (failover | roundRobin);
    failback-timer failback-timer;
    retry-interval retry-interval;
    maximum-queue-length maximum-queue-length;
    bind-address bind-address;
    udp-port udp-port;
    feid-mso-data feid-mso-data;
    feid-mso-domain-name feid-mso-domain-name;
    trusted-element;
    default-peer default-peer;
}
```

To configure an RKS plug-in:

1. From configuration mode, access the configuration statements for RKS plug-ins. In this sample procedure, west-region is the name of the SAE group, and rksPlugin is the name of the plug-in.

   user@host# **edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks**

2. Specify the mode for load-balancing RKSs.

   [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks]
   user@host# **set load-balancing-mode** (failover | roundRobin)

3. Specify if and when the SAE attempts to fail back to the default peer.

   [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks]
   user@host# **set failback-timer** *failback-timer*

4. Specify the time the SAE waits for a response from an RKS before it resends the packet.

   [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks]
   user@host# **set retry-interval** *retry-interval*

5. Specify the maximum number of unacknowledged messages that the plug-in receives from the RKS before it discards new messages.

   [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks]
   user@host# **set maximum-queue-length** *maximum-queue-length*

6. (Optional) Specify the source IP address that the plug-in uses to communicate with the RKS.

   [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks]
   user@host# **set bind-address** *bind-address*

7. (Optional) Specify the source UDP port or a pool of ports that the plug-in uses to communicate with the RKS.

   [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks]
   user@host# **set udp-port** *udp-port*

8. (Optional) Specify the multiple service operator (MSO)—defined data in the financial entity ID (FEID) attribute, which is included in event messages.

   [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks]
   user@host# **set feid-mso-data** *feid-mso-data*

9. (Optional) Specify the MSO domain name in the FEID attribute that uniquely identifies the MSO for billing and settlement purposes.

   [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks]
   user@host# **set feid-mso-domain-name** *feid-mso-domain-name*

10. (Optional) When the SAE is running as a policy server—which means that the SAE sends event messages directly to the RKS—enable the SAE as a trusted network element.

    [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks]
    user@host# **set trusted-element**

11. Specify the name of the primary RKS peer to which the SAE sends accounting packets.

    See *Configuring Record-Keeping Server Peers for Plug-Ins* on page 59.

    [edit shared sae group west-region configuration plug-ins pool rksPlugin pcmm-rks]
    user@host# **set default-peer** *default-peer*

12. (Optional) Verify your RKS plug-in configuration.

```
[edit shared sae group west-region configuration plug-ins pool rksPlugin
pcmm-rks]
user@host> show
load-balancing-mode failover;
failback-timer -1;
retry-interval 3000;
maximum-queue-length 10000;
feid-mso-domain-name abcd.com;
trusted-element;
default-peer radius01;
```

13. (Optional) Specify an RKS plug-in for specific CMTS devices.

    See *Configuring CMTS-Specific RKS Plug-Ins* on page 63.

Related Information

For additional information, see the following source:

- For information about setting up SAE groups, see *SDX Getting Started Guide, Chapter 16, Setting Up an SAE with the SRC CLI*.

Configuring CMTS-Specific RKS Plug-Ins

You can configure an RKS plug-in for specific CMTS devices. When there are events for the CMTS device, the SAE sends the events to the specified plug-in.

Use the following configuration statement to assign a CMTS-specific RKS plug-in.

```
shared sae configuration driver pcmm cmts-specific-rks-plug-ins name {
    rks-plug-in rks-plug-in;
}
```

To configure a CMTS-specific RKS plug-in:

1. From configuration mode, access the configuration statements for RKS plug-ins. In this sample procedure, west-region is the name of the SAE group, and cmtsPlugin is the name of the plug-in assignment.

   user@host# **edit shared sae group west-region configuration driver pcmm cmts-specific-rks-plug-ins cmtsPlugin**

2. Specify the name of the CMTS-specific RKS plug-in.

   [edit shared sae group west-region configuration driver pcmm cmts-specific-rks-plug-ins cmtsPlugin]
   user@host# **set rks-plug-in** *rks-plug-in*

3. (Optional) Verify your configuration.

   ```
   [edit shared sae group west-region configuration driver pcmm
   cmts-specific-rks-plug-ins cmtsPlugin]
   user@host# show
   rks-plug-in rksPlugin;
   ```

Related Information

For additional information, see the following source:

- For information about setting up SAE groups, see *SDX Getting Started Guide, Chapter 16, Setting Up an SAE with the SRC CLI*.

Chapter 6

Configuring the SAE for a PCMM Environment with SDX Configuration Editor

This chapter how to set up the SAE for a PCMM environment on a Solaris platform with SDX Configuration Editor. You can also use the SRC CLI to configure the SAE on a C-series platform or a Solaris platform. See *Chapter 5, Configuring the SAE for a PCMM Environment with the SRC CLI*.

Topics in this chapter include:

- Overview of Configuring the SAE for a Cable Network Environment on page 65
- Configuring the SAE to Manage PCMM Devices with SDX Configuration Editor on page 66
- Setting Up SAE Communities on page 69
- Configuring SAE Properties for the Event Notification API on page 70
- Configuring PCMM Record-Keeping Server Plug-Ins on page 71
- Configuring RKS Peers on page 74
- Configuring CMTS-Specific RKS Plug-Ins on page 75

Overview of Configuring the SAE for a Cable Network Environment

The tasks to configure the SAE for a cable network environment are:

1. Configuring the SAE to Manage PCMM Devices with SDX Configuration Editor on page 66.
2. Configuring the Session Store Feature. See *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 3, Configuring the SAE with SDX Configuration Editor*.
3. Setting Up SAE Communities on page 69.
4. Configuring SAE Properties for the Event Notification API on page 70 (if you are using an external address manager).

5. Configuring PCMM Record-Keeping Server Plug-Ins on page 71 (if you are using the SAE's embedded policy server).

6. Configuring RKS Peers on page 74.

7. Configuring CMTS-Specific RKS Plug-Ins on page 75

In addition to configuring the SAE, you need to:

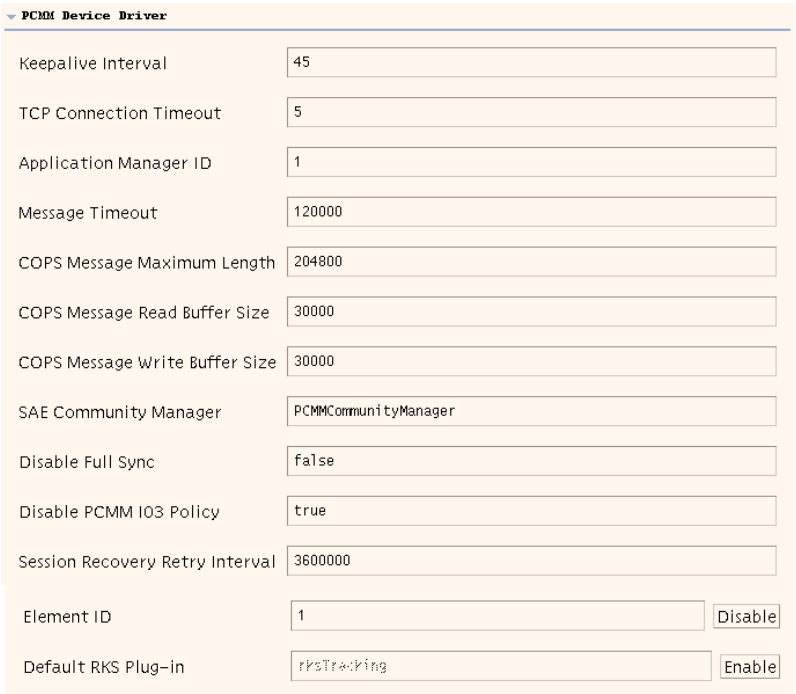
1. Configure the CMTS device in the directory (if you are using the SAE's embedded policy server). See *Adding Objects for CMTS Devices to the Directory with SDX Admin* on page 81.

2. Configure the NIC (if you are using assigned IP subscribers). See *Chapter 9, Using the NIC Resolver in a PCMM Environment*.

3. Enable the COPS interface on the CMTS device. See the documentation for your CMTS device for information about how to do this.

Configuring the SAE to Manage PCMM Devices with SDX Configuration Editor

The SAE connects to the PCMM device by using a COPS over TCP connection. The PCMM device driver controls this connection. You create a PCMM device driver for each CMTS device that the SAE manages. You can specify properties for the PCMM device driver in the Router tab of SDX Configuration Editor.



Keepalive Interval [s]

- Interval between keepalive messages sent from the COPS client (the PCMM device) to the COPS server (the SAE). The COPS client monitors the COPS connection by sending keepalive messages at random intervals between one-fourth and three-fourths of the specified interval. If the client or the server does not receive the expected keepalive answer within the specified timeout, the client closes the connection.
- Value—Number of seconds in the range 0–2147483647. A value of 0 means that the timeout is disabled.
- Default—45
- Property name—Router.pcmm.keepalive

TCP Connection Timeout [s]

- Timeout for opening a TCP connection to the PCMM device.
- Value—Number of seconds in the range 0–2147483647
- Default—5
- Property name—Router.pcmm.open_connection_timeout

Application Manager ID

- When this SAE is configured as the application manager, the identifier of the application manager. The application manager includes this identifier in all messages that it sends to the policy server. The policy server passes this ID to the CMTS device in Gate Control messages. The CMTS device returns the ID associated with the gate to the policy server. The policy server uses this information to associate gate messages with a particular application manager.
- Value—4-byte unsigned integer; must be unique in a service provider network
- Default—0

Message Timeout [ms]

- Amount of time that the COPS server (the SAE) waits for a response to COPS requests from the COPS client (the PCMM device). Under a high load the PCMM device may not be able to respond fast enough to COPS requests. Change this value only if a high number of COPS timeout events appear in the error log.
- Value—Number of milliseconds in the range 0–2147483647
- Default—120000
- Property name—Router.pcmm.message_timeout

COPS Message Maximum Length [bytes]

- Maximum length of a COPS message.
- Value—Number of bytes in the range 4 bytes to 2 GB
- Guidelines—We recommend that you use the default setting.
- Default—204800
- Property name—Router.pcmm.message_max_length

COPS Message Read Buffer Size [bytes]

- Buffer size for receiving COPS messages from the COPS client.
- Value—Number of bytes in the range 4 bytes to 2 GB
- Guidelines—We recommend that you use the default setting unless you are instructed to change it by Juniper Networks engineers.
- Default—30000
- Property name—Router.pcmm.message_read_buffer_size

COPS Message Write Buffer Size [bytes]

- Buffer size for sending COPS messages to the COPS client.
- Value—Number of bytes in the range 4 bytes to 2 GB
- Guidelines—We recommend that you use the default setting unless you are instructed to change it by Juniper Networks engineers.
- Default—30000
- Property name—Router.pcmm.message_write_buffer_size

SAE Community Manager

- Name of the community manager that manages PCMM driver communities. Active SAEs are selected from this community. You define community managers in the Ext. Interfaces tab of SDX Configuration Editor. See *Configuring the SAE Community Manager* on page 69.
- Value—Community name
- Default—PCMMCommunityManager
- Property name—Router.pcmm.community.name

Session Recovery Retry Interval

- Time between attempts by the SAE to restore service sessions that are being recovered in the background when state synchronization completes with a state-data-incomplete error. The SAE attempts to restore a service session if it receives a service modification or deactivation request for an unrecovered service session before the next interval.
- Value—Number of milliseconds in the range 0–2147483647
- Guidelines—We recommend setting this value to 3600000 (1 hour) or longer.
- Default—3600000

Element ID

- Enables or disables and sets the unique identifier that the SAE uses to identify itself when it originates RKS events.
- Value—8-byte unsigned integer in the range 0–99999; must be unique within a PCMM network
- Default—0
- Property name—Router.pcmm.emid

Default RKS Plug-In

- Enables or disables and sets the RKS plug-in to which the SAE sends event messages.
- Value—Name of an RKS plug-in; see *Configuring PCMM Record-Keeping Server Plug-Ins* on page 71
- Default—Disabled
- Property name—Router.pcmm.rks.plugin

Setting Up SAE Communities

You can configure the following for SAE communities:

- Define the members of an SAE community by adding the IP addresses of SAEs in the community to the virtual router object of the network device in the directory.

  See *Creating a Virtual Router for the CMTS Device with SDX Admin* on page 83.

- Configure parameters for the SAE community manager.

  See *Configuring the SAE Community Manager* on page 69.

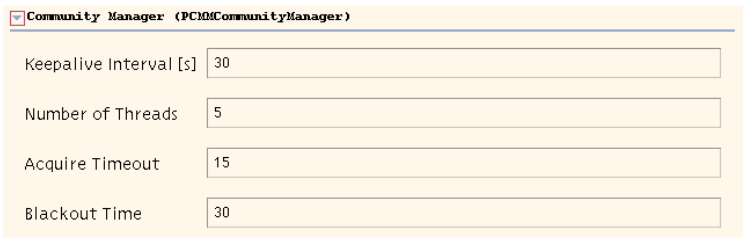
- Specify the name of the community manager in the PCMM driver configuration.

  See *Configuring the SAE to Manage PCMM Devices with SDX Configuration Editor* on page 66.

- If there is a firewall in the network, configure the firewall to allow SAE messages through.

Configuring the SAE Community Manager

You can configure the properties for an SAE community manager in the Ext. Interface tab of SDX Configuration Editor.



Keepalive Interval [s]

- Interval between keepalive messages sent from the active SAE to the passive members of the community.
- Value—Number of seconds in the range 0–2147483647
- Default—30
- Property name—SAEFeature.PCMMCommunityManager.heartbeat

Number of Threads

- Number of threads that are allocated to manage the community.
- Value—Integer in the range 0–2147483647
- Guidelines—You generally do not need to change this property.
- Default—5
- Property name—SAEFeature.PCMMCommunityManager.num_threads

Acquire Timeout

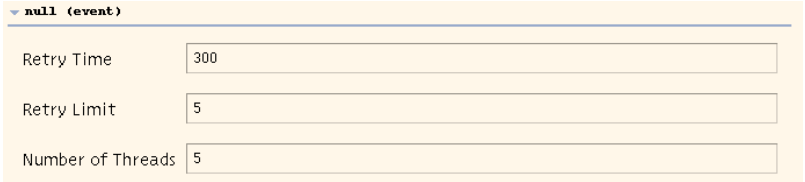
- Amount of time an SAE waits for a remote member of the community when it is acquiring a distributed lock. To avoid race conditions when the SAE community is determining which SAE is the active SAE, the community manager has a distributed lock. When an SAE attempts to become the active SAE, it needs to acquire the distributed lock.
- Value—Number of seconds in the range 0–2147483647
- Guidelines—You generally do not need to change this property.
- Default—15
- Property name—SAEFeature.PCMMCommunityManager.acquire_timeout

Blackout Time

- Amount of time that an active SAE must wait after it shuts down before it can try to become the active SAE of the community again.
- Value—Number of seconds in the range 0–2147483647
- Default—30
- Property name—SAEFeature.PCMMCommunityManager.blackout_time

Configuring SAE Properties for the Event Notification API

You can configure the SAE properties for the event notification API in the Ext. Interface tab of SDX Configuration Editor.



Retry Time

- Amount of time between attempts to send events that could not be delivered.
- Value—Number of seconds in the range 0–2147483647
- Default—300
- Property name—SAEFeature.event.retry_time

Retry Limit

- Number of times an event fails to be delivered before the event is discarded.
- Value—Integer in the range 0–2147483647
- Default—5
- Property name—SAEFeature.event.retry_limit

Number of Threads

- Number of threads allocated to process events.
- Value—Integer in the range 0–2147483647
- Default—5
- Property name—SAEFeature.event.num_threads

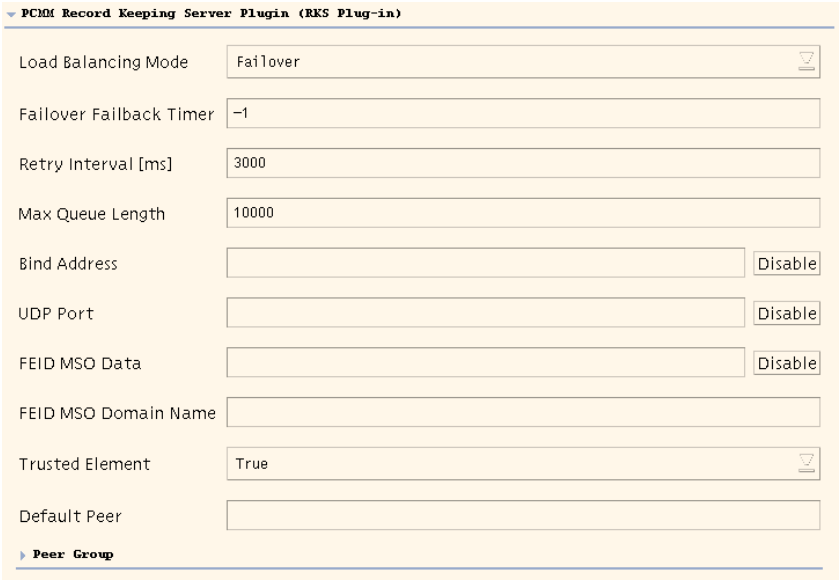
Configuring PCMM Record-Keeping Server Plug-Ins

You configure PCMM RKS plug-ins in the Plug-Ins tab of SDX Configuration Editor. To set up PCMM record-keeping server plug-ins:

1. In the navigation pane, select the SAE object for which you want to configure plug-ins.
2. Select the Plug-Ins tab.

   The Plug-Ins pane appears.
3. In the Plug-In Pool area of the Plug-Ins pane, select **PCMM Record Keeping Server Plugin** from the drop-down list, and click **Create a New Instance of**.

   The instance appears in the Plug-In Pool area.



4. Fill in the plug-in instance fields as described below.
5. In the Peer Group area, create at least one peer to use as the default peer. See *Configuring RKS Peers* on page 74.
6. In the PCMM Device Driver configuration, add the RKS plug-in as the default RKS plug-in or as a CMTS-specific plug-in. See *Configuring the SAE to Manage PCMM Devices with SDX Configuration Editor* on page 66.

Load Balancing Mode

- Selects the mode for load-balancing RKSs.
- Value—Failover, round-robin
  - Failover—SAE sends requests to the RKS configured as the default peer. If the default peer fails, the SAE uses the next server configured in the peer group. The SAE cycles through the configured servers as needed.
  - Round-robin—SAE alternates requests between all RKSs configured in the peer group.
- Default—Failover
- Property name—loadBalancingMode

Failover failback timer

- Controls if and when the SAE attempts to fail back to the default peer.
- Value—Integer
  - *n*—Number of seconds after a failover that the SAE attempts to fail back; range is 1–2147483647
  - 0—SAE always attempts to fail back
  - –1—SAE never attempts to fail back

- Default—–1
- Property name—failbackTimer

Retry interval [ms]

- Time the SAE waits for a response from an RKS before it resends the packet. The SAE keeps sending packets until either the RKS acknowledges the packet or the maximum timeout is reached.
- Value—Number of milliseconds in the range 0–2147483647
- Default—3000
- Property name—local.retryInterval

Max Queue Length

- Maximum number of unacknowledged messages that the plug-in receives from the RKS before it discards new messages.
- Value—Integer in the range 0–2147483647
- Default—10000
- Property name—local.maxWaitingQueueLength

Bind Address

- Source IP address that the plug-in uses to communicate with the RKS.
- Value—IP address; if you do not specify an address, the global default address is used. The SAE automatically sets the global default address when you run the **etc/config** command during initial configuration of the SAE. The property for the global address is the AccountingMgr.local.address property in the */opt/UMC/sae/etc/default.properties* file.
- Default—No value
- Property name—local.address

UDP Port

- Source UDP port or a pool of ports that the plug-in uses to communicate with the RKS.
- Value—You can enter a single port number, a pool of port numbers, or a list of port numbers and port ranges. If you do not specify a UDP port, the global default port is used (see *SDX Subscribers and Subscriptions Guide, Chapter 5, Configuring Subscriber-Related Properties on the SAE on a Solaris Platform*).
  - Port number in the range 1–65535
  - A range of ports in the format port-port; for example, 7000-7003
  - A comma-separated list of port numbers and port ranges
- Default—No value
- Example—7000-7003, 7006, 7007-7009
- Property name—local.port

FEID MSO Data

- MSO-defined data in the financial entity ID (FEID) attribute, which is included in event messages.
- Value—First eight bytes of the FEID attribute
- Default—Zero filled
- Property name—feid.msoData

FEID MSO Domain Name

- MSO domain name in the FEID attribute that uniquely identifies the MSO for billing and settlement purposes.
- Value—Domain name up to 239 bytes; begins at the ninth byte of the FEID attribute
- Default—No value
- Property name—feid.msoDomainName

Trusted Element

- When the SAE is running as a policy server—which means that the SAE sends event messages directly to the RKS—specifies whether or not the SAE is a trusted network element.
- Value
  - True—The SAE is a trusted element.
  - False—The SAE is not a trusted element.
- Default—True
- Property name—trusted

Default peer

- Name of the primary RKS to which the SAE sends accounting packets.
- Value—Name of the RKS as defined in the RKS peer configuration
- Default—No value
- Property name—defaultPeer

Configuring RKS Peers

An RKS peer is an instance of an RKS. A PCMM environment has a primary RKS and optionally a secondary RKS. The primary RKS is mandatory, and you assign the RKS as primary by configuring it as the default peer in the RKS plug-in. The secondary RKS is optional, and it is an RKS peer that is not configured as the default peer. If you define multiple nondefault peers, one of them is randomly chosen to be the secondary RKS.

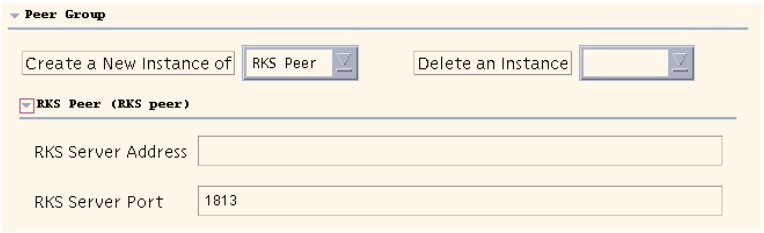
RKS peers are configured in the peer group for each PCMM RKS plug-in instance. To create an RKS peer:

1. In the Peer Group area of a PCMM RKS plug-in instance, select RKS Peer and click Create a New Instance of.

   The Create New Instance dialog box appears.

2. Assign a name to the instance, and click **OK**.

   The new peer appears in the Peer Group area.



3. Fill in the fields as described below.

RKS Server Address

- IP address of the RKS server to which the SAE sends accounting data.
- Value—IP address
- Default—No value
- Property name—peer.<peer name>.remote.address

RKS Server Port

- Port used for sending accounting packets.
- Value—Valid UDP port
- Default—1813
- Property name—peer.<peer name>.remote.port

Configuring CMTS-Specific RKS Plug-Ins

You can configure an RKS plug-in for specific CMTS devices. When there are events for the CMTS device, the SAE sends the events to the specified plug-in.

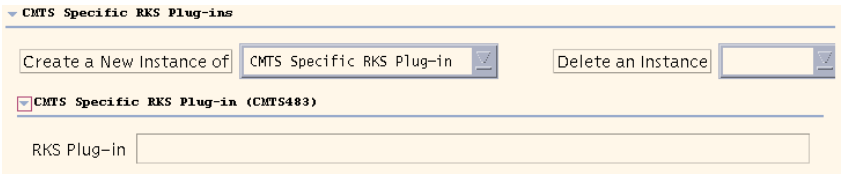
To assign a CMTS-specific RKS plug-in:

1. In the CMTS Specific RKS Plug-ins area of a PCMM device driver configuration, select CMTS Specific RKS Plug-in, and click **Create a New Instance of**.

   The Create New Instance dialog box appears.

2. Assign the name of the CMTS device as the instance name, and click **OK**.

The new plug-in instance appears.



3. Fill in the RKS Plug-in field.

RKS Plug-in

- Name of the plug-in to which the SAE sends events for this CMTS device.
- Value—Name of an RKS plug-in. See *Configuring PCMM Record-Keeping Server Plug-Ins* on page 71.
- Default—No value
- Property name—Router.pcmm. < CMTS name > .rks.plugin

Chapter 7

Adding Objects for CMTS Devices with the SRC CLI

This chapter describes how to configure objects for cable modem termination system (CMTS) devices with the SRC CLI. You can also use SDX Admin to configure objects on a Solaris platform. See *Chapter 8, Adding Objects to the Directory with SDX Admin*.

Topics in this chapter include:

- Adding Objects for CMTS Devices with the SRC CLI on page 77
- Creating Virtual Routers for the CMTS Device with the SRC CLI on page 78

Adding Objects for CMTS Devices with the SRC CLI

To manage CMTS devices, the SAE creates and manages pseudointerfaces that it associates with a virtual router object. Each CMTS device in the SRC network must appear in the configuration as a router object, and it must be associated with a virtual router object called default. The router and virtual router are not actually configured on the CMTS device; the router and virtual router provide a way for the SAE to manage the CMTS device by using the SAE's embedded policy server.

Use the following configuration statements to add a router object:

```
shared network device name {
    description description;
    management-address management-address;
    device-type (junose | junos | pcmm | proxy);
    qos-profile [qos-profile...];
}
```

To add a router:

1. From configuration mode, access the configuration statements that configure network devices. In this sample procedure, pcmm_dtr is the name of the object.

   user@host# **edit shared network device pcmm_dtr**

2. (Optional) Add a description for the CMTS device.

   [edit shared network device pcmm_dtr]
   user@host# **set description** *description*

3. Add the IP address of the CMTS device.

   [edit shared network device pcmm_dtr]
   user@host# **set management-address** *management-address*

4. (Optional) Specify the type of device that you are adding.

   [edit shared network device pcmm_dtr]
   user@host# **set device-type pcmm**

5. (Optional) Verify your configuration.

```
[edit shared network device pcmm_dtr]
user@host# show
description "CMTS device";
management-address 192.168.3.5;
device-type pcmm;
interface-classifier {
  rule rule-0 {
    script #;
  }
}
```

Creating Virtual Routers for the CMTS Device with the SRC CLI

You need to add a virtual router object called default to the CMTS device. Use the following configuration statements to add a virtual router:

shared network device name virtual-router name {
    sae-connection [*sae-connection*...];
    snmp-read-community *snmp-read-community*;
    snmp-write-community *snmp-write-community*;
    scope [*scope*...];
    local-address-pools *local-address-pools*;
    static-address-pools *static-address-pools*;
    tracking-plug-in [*tracking-plug-in*...];
}

To add a virtual router:

1. From configuration mode, access the configuration statements for virtual routers. In this sample procedure, pcmm_dtr s the name of the router and default as the name of the virtual router.

   user@host# **edit shared network device pcmm_dtr virtual-router default**

2. Specify the addresses of SAEs that can manage this router. This step is required for the SAE to work with the router.

   [edit shared network device pcmm_dtr virtual-router default]
   user@host# **set sae-connection** [*sae-connection*...]

3. (Optional) Specify an SNMP community name for SNMP read-only operations for this VR.

   [edit shared network device pcmm_dtr virtual-router default]
   user@host# **set snmp-read-community** *snmp-read-community*

4. (Optional) Specify an SNMP community name for SNMP write operations for this virtual router.

   [edit shared network device pcmm_dtr virtual-router default]
   user@host# **set snmp-write-community** *snmp-write-community*

5. (Optional) Specify service scopes assigned to this virtual router.

   See *SDX Services and Policies Guide, Chapter 1, Managing Services with the SRC CLI*.

   [edit shared network device pcmm_dtr virtual-router default]
   user@host# **set scope** [*scope*...]

6. (Optional) Specify the list of IP address pools that a CMTS virtual router currently manages and stores.

   If you are using assigned IP subscribers along with the network information collector (NIC), you need to configure either a local or static address pool so that the NIC can resolve the IP-to-SAE mapping.

   [edit shared network device pcmm_dtr virtual-router default]
   user@host# **set local-address-pools** *local-address-pools*

7. (Optional) Specify the list of IP address pools that a CMTS VR manages but does not store.

   If you are using assigned IP subscribers along with the NIC, you need to configure either a local or static address pool so that the NIC can resolve the IP-to-SAE mapping.

   [edit shared network device pcmm_dtr virtual-router default]
   user@host# **set static-address-pools** *static-address-pools*

8. (Optional) Specify the plug-ins that track interfaces that the SAE manages on this virtual router.

   [edit shared network device pcmm_dtr virtual-router default]
   user@host# **tracking-plug-in** [*tracking-plug-in*...]

9. (Optional) Verify your configuration.

```
[edit shared network device pcmm_dtr virtual-router default]
user@host# show
sae-connection [ 10.14.39.2 10.10.5.30 ];
snmp-read-community ********;
snmp-write-community ********;
scope POP-Westford;
local-address-pools "10.25.8.0 10.25.20.255";
tracking-plug-in rksPlugin;
```

Chapter 8

Adding Objects to the Directory with SDX Admin

This chapter describes how to configure objects for CMTS devices with SDX Admin. You can also use the SRC CLI to configure objects on a C-series platform or a Solaris platform. See *Chapter 7, Adding Objects for CMTS Devices with the SRC CLI*.

This chapter contains the following topics:

- Adding Objects for CMTS Devices to the Directory with SDX Admin on page 81
- Creating a Virtual Router for the CMTS Device with SDX Admin on page 83

Adding Objects for CMTS Devices to the Directory with SDX Admin

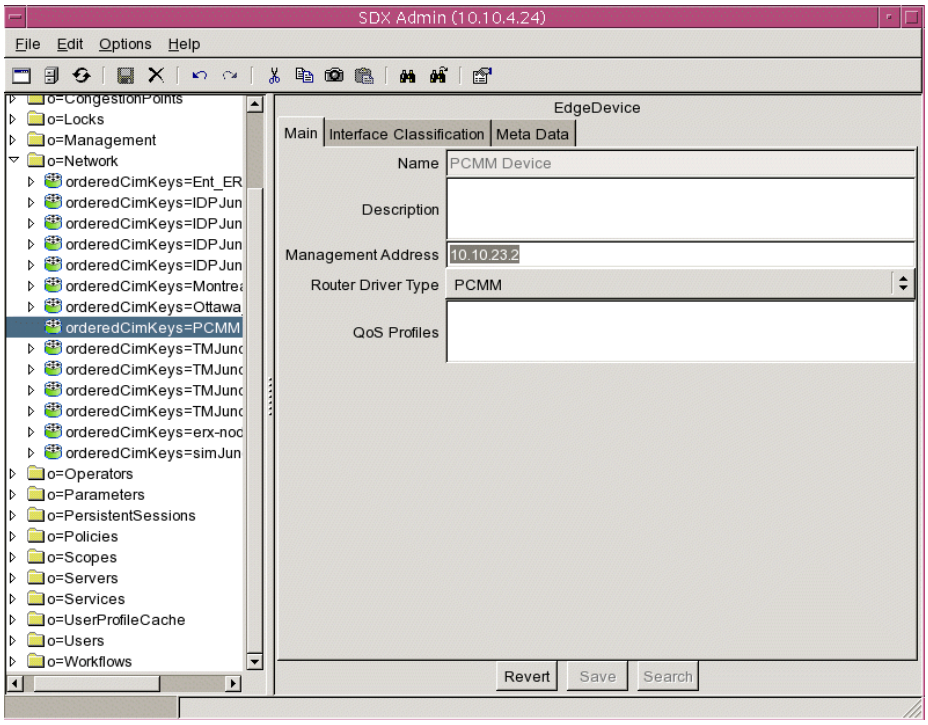
To manage CMTS devices, the SAE creates and manages pseudointerfaces that it associates with a virtual router object in the directory. Each CMTS device in the SRC network must appear in the directory as a router object, and it must be associated with a virtual router object called default. The router and virtual router are not actually configured on the CMTS device; the router and virtual router in the directory provide a way for the SAE to manage the CMTS device by using the SAE's embedded policy server.

To add a CMTS device to the directory with SDX Admin:

1. In the navigation pane, highlight *o = Network*, and right-click.
2. Select **New > EdgeDevice**.

   The New EdgeDevice dialog box appears.
3. In the New EdgeDevice dialog box, enter the name of the CMTS device, and click **OK**.

The name of the new device appears in the navigation pane, and information about the device appears in the EdgeDevice pane.



4. Set the parameters in the Main tab of the EdgeDevice pane.
5. Click **Save** in the EdgeDevice pane.
6. Create a virtual router for the CMTS device. See *Creating a Virtual Router for the CMTS Device with SDX Admin* on page 83.

Description

- Information about this device; keywords that the SDX Admin find utility uses.
- Value—Text string
- Default—No value

Management Address

- IP address of the CMTS device. The SAE uses this address to establish a COPS connection with the CMTS device.
- Value—IP address
- Default—No value

Router Driver Type

- Type of device that this directory object will be used to manage.
- Value
  - JUNOSe—JUNOSe router
  - JUNOS—JUNOS routing platform
  - PCMM—PCMM-compliant CMTS device

  If you do not fill in this field, the device driver ignores this router driver.
- Default—No value

QoS Profiles

- For JUNOSe routers only, QoS profiles that are configured on the router.
- Value—List of QoS profiles on separate lines
- Example—atm-default
- Default—No value

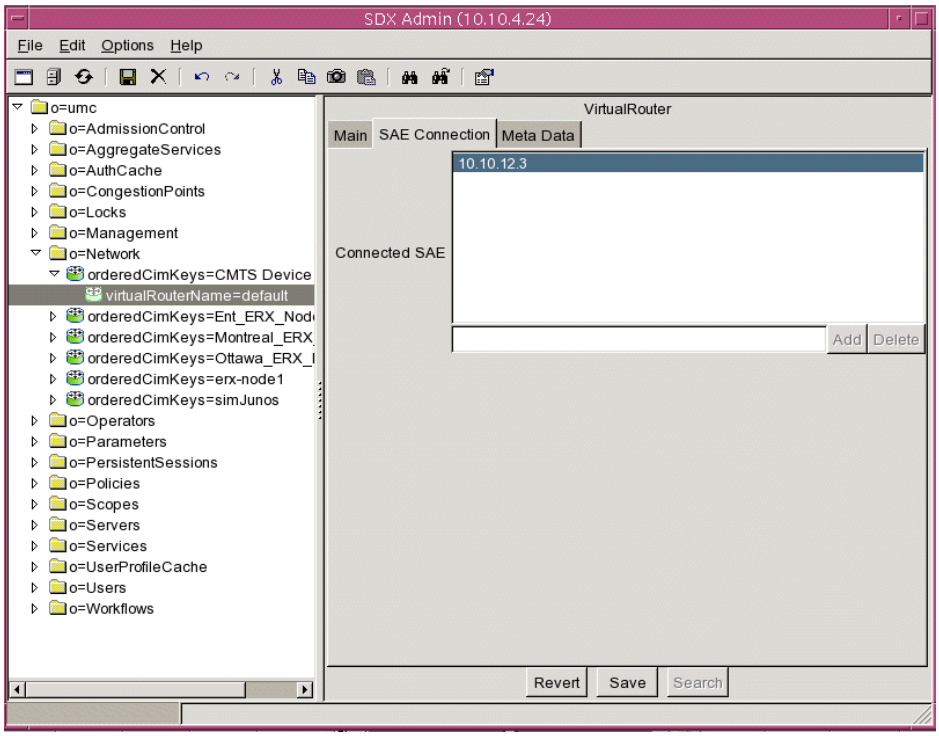
Creating a Virtual Router for the CMTS Device with SDX Admin

You need to add a virtual router object called default to the CMTS device. To add a virtual router with SDX Admin:

1. In the navigation pane, right-click on the CMTS device.
2. Select **New > VirtualRouter**.

   The New EdgeDevice dialog box appears.
3. In the New VirtualRouter dialog box, enter the name default, and click **OK**.

The default virtual router appears in the navigation pane, and information about the virtual router appears in the VirtualRouter pane.



4. Configure virtual router parameters in the Main Tab. See *Configuration Parameters for Virtual Routers* on page 84.

5. Select the SAE Connection tab of the VirtualRouter pane, and add SAEs that are connected to the CMTS device. This list becomes the community of SAEs.

   To add an SAE:

   a. Type the IP address of the SAE in the field below the Connected SAE box.

   b. Click **Add**.

6. Click **Save** in the VirtualRouter pane.

Configuration Parameters for Virtual Routers

Use the fields in this section to define virtual router objects. If you are using assigned IP subscribers along with the NIC, you need to configure either a local or static address pool so that the NIC can resolve the IP-to-SAE mapping.

SNMP Read Community

- SNMP community name associated with SNMP read-only operations for this VR.
- Value—Text string
- Example—admin

SNMP Write Community

- SNMP community name associated with SNMP write operations for this VR.
- Value—Text string
- Example—public

Scope

- Service scopes assigned to this VR—See *Configuring Service Scopes* in *SDX Services and Policies Guide, Chapter 2, Managing Services on a Solaris Platform*.
- Value—Text string
- Example—POP-Westford

Local Address Pools

- List of IP address pools that the VR currently manages and stores. You must configure either a local address pool or a static address pool so that the NIC can resolve the IP-to-SAE mapping.
- Value—List of IP address pools. You can specify an unlimited number of IP address pools. You can specify either the first and last addresses in a range, or you can specify a subnet address, a subnet mask, and a list of addresses to exclude from the subnet.

  The IP pool syntax has the following format:

  ([< ipAddressStart > < ipAddressEnd >] |
  { < ipBaseAddress > /(< mask > | < digitNumber >)(, < ipAddressExclude >)* })

  where:

  - < ipAddressStart > —First IP address (version 4 or 6) in a range
  - < ipAddressEnd > —Last IP address (version 4 or 6) in a range
  - < ipBaseAddress > —Network base address
  - < mask > —Subnet mask
  - < digitNumber > —Integer specifying the length of the subnet mask
  - < ipAddressExclude > —List of IP addresses to be excluded from the subnet
  - |—Choice of expression; choose either the expression to the left or the expression to the right of this symbol
  - *—Zero or more instances of the preceding group

  You can use spaces in the syntax only to separate the first and last explicit IP addresses in a range.
- Default—No value
- Example—([10.10.10.5 10.10.10.250] {10.20.20.0/24})

Static Address Pools

- List of IP address pools that the VR manages but does not store. You can configure these address pools only in the SRC software. You must configure either a local address pool or a static address pool so that the NIC can resolve the IP-to-SAE mapping.
- Value—See the field Local Address Pools.
- Default—No value
- Example—([10.10.10.5 10.10.10.250] {10.20.20.0/24})

Managing SAE IOR

- Common Object Request Broker Architecture (CORBA) reference for the SAE managing this VR.
- Value—One of the following items:
  - The actual CORBA reference for the SAE
  - The absolute path to the interoperable object reference (IOR) file
  - A corbaloc URL in the form corbaloc::<host>:8801/SAE
    - <host> is the name or IP address of the SAE host
- Guidelines—The **PoolPublisher** and **IorPublisher** router initialization scripts provide this information when the router connects to the SAE. For information about configuring router initialization scripts, see *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 6, Using JUNOSe Routers in the SRC Network with a Solaris Platform* or *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 8, Using JUNOS Routing Platforms in the SRC Network with a Solaris Platform*. If you do not select one of these router initialization scripts, enter a value in this field.
- Default—No value
- Example—One of the following items:
  - Absolute path— */opt/UMC/sae/var/run/sae.ior*
  - corbaloc URL—boston:8801/sae
  - Actual IOR—IOR:000000000000002438444C3A736D67742E6A756E697...

Tracking Plug-in

- Plug-in instances that track interfaces that the SAE manages on this VR. The SAE calls these plug-ins after an interface comes up, when new policies are installed on the interface, and when the interface goes down.
- Value—Comma-separated list of plug-in instances
- Guidelines—Enter plug-in instances and NIC SAE plug-in agents that are specific to this VR. For information about configuring tracking plug-ins, see *SDX Subscribers and Subscriptions Guide, Chapter 5, Configuring Subscriber-Related Properties on the SAE on a Solaris Platform*.
- Default—No value
- Example—nicsae, flexRadius

Configuring SAE Communities

You define SAE communities by entering the SAEs in a community in the connected SAE field of the virtual router object.

When you modify a community, wait for passive session stores on the new community members to be updated before you shut down the current active SAE. Otherwise, if you add a new member to a community, and then a failover from the current active SAE to the new member is triggered immediately, the new member's session store may not have received all data from the active SAE's session store.

To define a community, select the SAE Connection tab of the VirtualRouter pane, and add the addresses of SAEs that can manage this CMTS device.

To add an SAE:

1. Type the IP address of the SAE in the field below the Connected SAE box.
2. Click **Add**.

To modify an SAE address:

1. Double-click the IP address of the SAE in the Connected SAE box.
2. Modify the IP address in the field below the Connected SAE box.
3. Click **Modify**.

To delete an SAE address:

1. Double-click the IP address of the SAE in the Connected SAE box.
2. Remove the IP address from the field below the Connected SAE box.
3. Click **Delete**.

Connected SAE

- SAEs that are connected to the CMTS device.
- Value—IP addresses
- Default—No value

Chapter 9

Using the NIC Resolver in a PCMM Environment

This chapter describes

- Overview of Using the NIC Resolver in a PCMM Environment on page 89
- Accessing the OnePopDynamicIp Configuration with the SRC CLI on page 89
- Accessing the OnePopDynamicIp Configuration on a Solaris Platform on page 90

Overview of Using the NIC Resolver in a PCMM Environment

If you are using the assigned IP subscriber method of logging in subscribers, and you are using the NIC to determine the subscriber's SAE, you need to configure a resolver on the NIC. The OnePopDynamicIp sample configuration data supports this scenario. The OnePopDynamicIp configuration supports one point of presence (POP) and provides no redundancy. The realm for this configuration accommodates the situation in which IP pools are configured locally on each virtual router object.

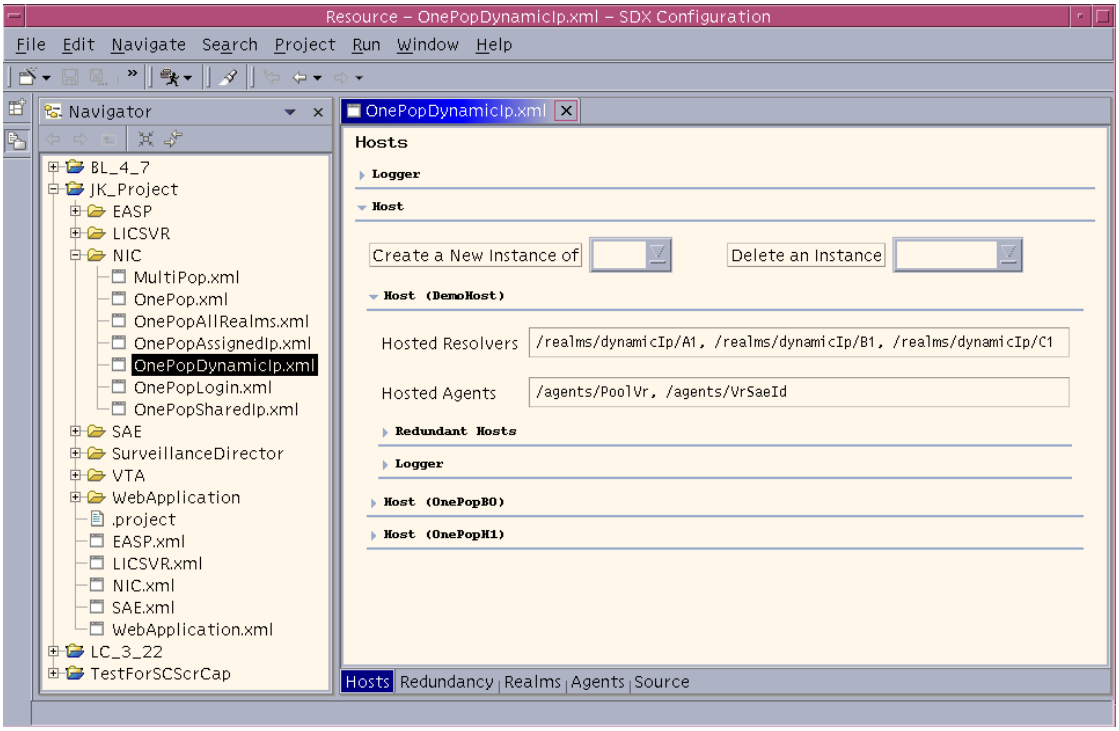
Accessing the OnePopDynamicIp Configuration with the SRC CLI

You can access the OnePopDynamicIp configuration in the SRC CLI. See *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 10, Configuring NIC with the SRC CLI* for information about configuring NIC scenarios with the SRC CLI.

Accessing the OnePopDynamicIp Configuration on a Solaris Platform

You can access the OnePopDynamicIp configuration in either SDX Admin or SDX Configuration Editor. Figure 16 shows the sample configuration in SDX Configuration Editor.

Figure 16: OnePopDynamicIP Sample Configuration in SDX Configuration Editor



For more information about the resolution process for OnePopDynamicIp, see *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 11, Configuring NIC on a Solaris Platform*.

Chapter 10

Using IPSec to Protect Communications Between the SAE and CMTS Device

This chapter describes the SRC application's support for the *PacketCable Multimedia Specification* (PCMM) as defined by Cable Television Laboratories, Inc. (CableLabs). It contains the following sections:

If you use the SRC software to manage a PCMM environment, IP security (IPSec) protects communications between the SAE and RADIUS and between the SAE and the CMTS device. The *PacketCable Multimedia Specification* outlines the security requirements for communication between components in a PCMM environment.

- Overview of IPSec on page 91
- IPSec Configuration for the SAE on page 93
- Before You Configure IPSec on page 94
- Protecting IPSec Configuration Properties on page 95
- Configuring IPSec for the SAE on page 95
- Configuring IPSec with SDX Configuration Editor on page 95
- Configuring IPSec on a Remote System on page 102
- Testing the IPSec Connection on page 103

Overview of IPSec

IPSec provides IP-level security for packets sent between specified hosts by using both authentication and encryption:

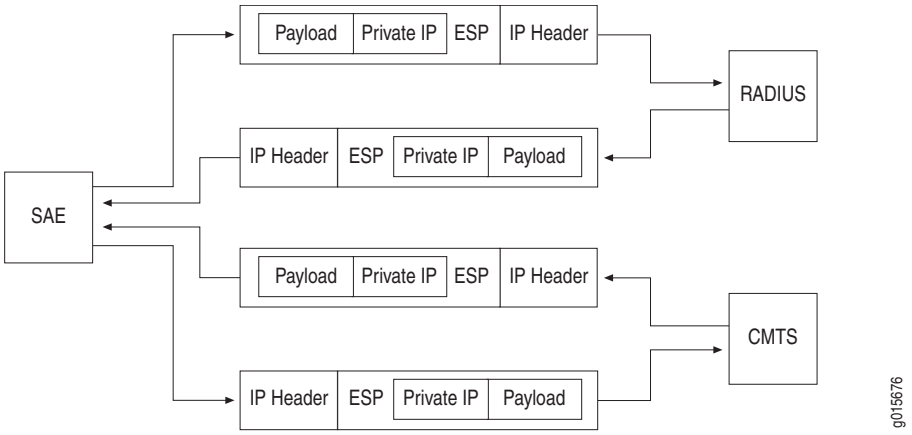
- Authentication ensures data integrity and verifies the identity of the sender and receiver.
- Encryption ensures data confidentiality; only the sender and intended recipient can read the information.

IPSec uses cryptographic keys during authentication and encryption. For authentication, the key and the data form a checksum value; for encryption, a key encrypts data before it is sent and decrypts data when it is received.

Before IPSec-protected communication can be established, both sender and receiver share configuration information with each other. As a result, IPSec defines a security association (SA), the set of security parameters that dictate how IPSec processes a packet, for a sender and for a receiver. These parameters include addressing and key information, both of which must be common to both hosts. Typically, a security association includes parameters for packets transmitted in one direction. Another security association is needed for packets transmitted in the opposite direction.

Figure 17 shows Encapsulating Security Payload (ESP) encapsulated packets sent between SAE and a RADIUS server, and between SAE and a CMTS device.

Figure 17: IPSec-Protected Communications



The SAE uses the IPSec implementation available on the Solaris platform on which the SAE runs. The SAE provides a configuration interface to simplify IPSec configuration for the SAE. For information about the IPSec implementation on the Solaris operating system, see the Sun product documentation at

http://docs.sun.com/app/docs/prod/solaris#hic

Security Keys

For a sender and receiver to participate in IPSec-protected communication, both must use the same type of key that is based on the algorithms used.

Key Types

IPSec uses different key algorithms for authentication and encryption. The SAE supports use of the following algorithms for authentication:

- Hashed Message Authentication Code using a Message Digest 5 key (HMAC-MD5)
- Hashed Message Authentication Code using a Secure Hash Standard 1 key (HMAC-SHA-1)

The SAE supports use of the following algorithms for encryption:

- Data Encryption Standard (DES)
- Triple Data Encryption Standard (3DES)
- Advanced Encryption Standard (AES)
- Blowfish

Which encryption algorithms are available depends on whether the system has the Solaris Encryption Kit installed. See the Solaris documentation for more information.

Key Management

The implementation of IPSec for the SAE uses automatic key management through Internet Key Management (IKE). IKE is a protocol that provides key generation and secure distribution. It also secures negotiations to create security associations.

The SAE configuration uses a preshared key for IKE negotiations. A preshared key is one whose value is shared by the administrators of the systems that participate in IPSec-protected communication. You define a value for the key and communicate the value of the key out-of-band to the system administrator who is configuring the CMTS device or RADIUS server. When you communicate the key value, make sure that only trusted parties have access to the key information.

NOTE: When you configure the value of this key for the SAE, you use SDX Configuration Editor. Anyone who can open SDX Configuration Editor can read the value for this key. The key value, however, is not stored in the LDAP directory.

Although SDX Configuration Editor supports only configuration of preshared keys, the Solaris operating system also supports certificate authentication. We recommend that you use preshared keys; however, you can configure certificate authentication directly from Solaris if required by your environment.

IPSec Configuration for the SAE

The SAE uses the IPSec implementation available on a system running the Solaris operating system version 5.9 or higher. These versions of the operating system support IKE.

SRC software configures basic IPSec parameters and provides a management interface in SDX Configuration Editor to simplify configuration tasks for properties specific to your environment. For example, the SAE configuration lets you configure the IP address to be used on the local host and the IP address to be used on the remote host for IPSec-protected traffic.

The basic IPSec configuration created by the SAE includes the following:

- IPv4 addressing—Supports IP addressing in the IPv4 format for local and remote identity types.
- Preshared keys—Lets you share key values between systems.
- Automatic key management through IKE—Manages security keys during negotiation of SAs.
- ESP—Provides confidentiality and authentication for each packet.
- IPSec transport mode—Specifies that ESP follow the IP header for a packet; ESP encapsulates the remainder of the packet.

Before You Configure IPSec

Before you start to configure IPSec for the SAE:

- Verify that the system on which the SAE is uses the Solaris operating system version 5.9 or higher.
- Verify which authentication algorithms and encryption algorithms are available on your Solaris platform.

  Which encryption algorithms are available depends on whether the system has the Solaris Encryption Kit installed. See the Solaris documentation for more information.
- Make sure that you are familiar with any configuration for IPSec present on the system running the SAE. If IPSec is already configured on the Solaris platform, make sure that system-wide policies are compatible with the IPSec configuration for SAE.

Before you start to configure IPSec from SDX Configuration Editor, collect the following information:

- Value of the preshared key

  Use a random key generator to obtain this value. To generate a random number, you can use the **od** command on a Solaris platform. See the Solaris documentation.
- Authentication algorithm to use
- Encryption algorithm to use
- IP address of the remote host
- (Optional) Port number to be used on the remote system

Protecting IPSec Configuration Properties

Make sure that a malicious user cannot obtain the IPSec configuration information. You can protect the configuration information by:

- Making configuration changes from the console of the terminal on which the SAE is running.
- Configuring SSH between the host from which you access the SAE and the host on which the SAE runs.

  See the documentation for these systems for information about setting up SSH between the hosts.

Configuring IPSec for the SAE

The procedure for configuring IPSec between the SAE and another application comprises the following steps:

1. Make sure that the authentication and encryption algorithms you plan to use are available on the local and remote hosts.
2. Configure IPSec on the system running the SAE.

   See *Configuring IPSec with SDX Configuration Editor* on page 95.
3. Configure IPSec on the remote system, such as a CMTS device or a RADIUS server.

   See the documentation for the remote system.
4. Test the IPSec connection. See the Solaris documentation.

NOTE: Before you activate the IPSec configuration, make sure that the IPSec configuration is working; otherwise, troubleshooting the IPSec configuration becomes very difficult.

Configuring IPSec with SDX Configuration Editor

You can use SDX Configuration Editor to configure IPSec properties required to protect traffic between the SAE and another system. For information about using SDX Configuration Editor, see *SDX Getting Started Guide, Chapter 39, Using SDX Configuration Editor*.

To configure IPSec attributes from SDX Configuration Editor:

1. In the navigation pane of SDX Configuration Editor, right-click an object, select **SDX System Configuration**, and then select **New Configuration File**.
2. In the Create a New Configuration File dialog box, enter a filename in the File Name field, select ipSec_conf in the Template field, and click **OK**.

3. In the navigation pane, double-click the name of the new file.

   The IPSec Transport Connections pane appears.

4. Click **Solaris Hosts** to expand it, select **Host** in the drop-down list box, click **Create a New Instance of**, and enter the Instance Name in the Create a New Instance dialog box.

   The new instance appears.

5. Configure host properties. Use the field descriptions in *Configuring Host Properties* on page 96 to configure the properties.

6. Expand IPSec Connections; then for each connection, select **Connection** in the drop-down list box, click **Create a New Instance of**, and enter the Instance Name in the Create a New Instance dialog box.

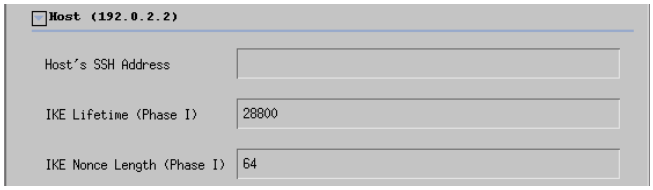
   The new connection instance appears.

7. Expand the Connection section for a specified connection, and enter field values. Use the field descriptions in *Configuring Connection Properties* on page 97 to configure the properties.

8. Expand the IPSec Details section for a specified connection, and enter field values. Use the field descriptions in *Configuring IPSec Properties to Establish Key Exchange and SAs* on page 99 to configure the properties.

Configuring Host Properties

Use the host properties area to define IPSec configuration properties for the Solaris platform.



Host's SSH Address

- IP address or hostname to be used for IPSec configuration on the Solaris platform.
- Value—IP address or fully qualified hostname used for IPSec configuration on the on the Solaris platform; can include the port for an SSH server.
- Default—No value
- Example

  IP address with port 22 for SSH—192.0.2.2:22

  Hostname—sae.company.com
- Property name—sshAddress

IKE Lifetime (Phase 1)

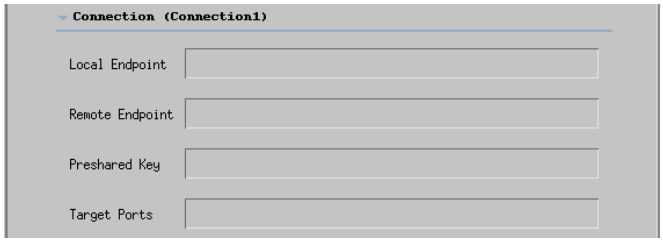
- Length of time phase 1 SA can be active for all IPSec connections on the Solaris platform.
- Value—Length of time in seconds
- Guidelines—We recommend a minimum lifetime of 28800 seconds (8 minutes).
- Default—28800
- Property name—ikeNonceLength

IKE Nonce Length (Phase 1)

- Size of the nonce token used during phase 1 of IKE negotiation.
- Value—Number of bytes in the range 1–64
- Guidelines—This property sets this value for all IPSec connections on the Solaris platform.
- Default—64
- Property name—ikeLifeTime

Configuring Connection Properties

Use the Connection properties area to define the source and destination for IPSec-protected communications, and the type of key to use in IKE negotiation.



Local Endpoint

- IP address for IPSec to use on the local Solaris platform on which the SAE is running.
- Value— < IP address >
- Guidelines—This is a required entry.
- Property name—localEndPt

Remote Endpoint

- IP address to use on the remote system.
- Value— < IP address >
- Guidelines—This is a required entry.
- Property name—RemoteEndPt

Preshared Key

- Value of the key to be shared between the SAE and the remote system. IKE negotiation uses this key.
- Value—A number in hexadecimal notation
- Guidelines—This is a required entry.

  The different IKE algorithms support keys of various lengths. In general, longer keys provide more security than shorter keys provide. The length of the key should comply with the security policies at your site.

  Protect the value of this key. Unauthorized access to the key value can compromise data that is protected by this key.

- Property name—presharedKey

Target Ports

- Well-known port numbers associated with applications that participate in IPSec-protected communications.
- Value—Port number associated with an application

  Blank—All port numbers

- Guidelines—This is a required entry.

  We recommend that the field remain blank to have IPSec protect all traffic between the local and remote systems.

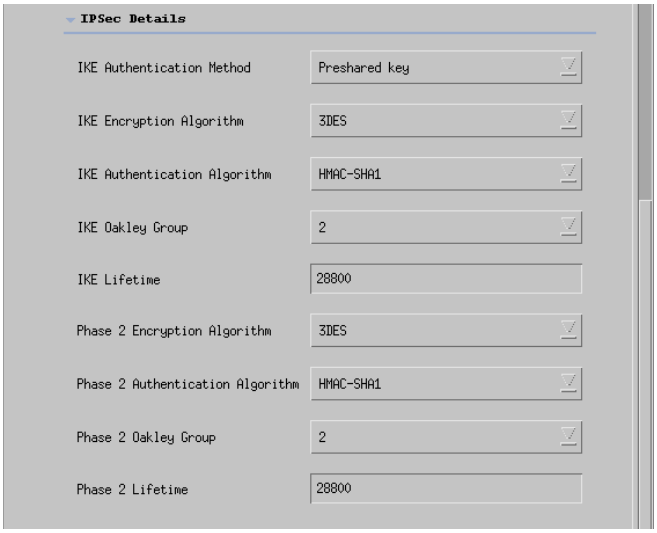
  If you specify port numbers, you can enter more than one port number, with commas separating the port numbers. The following list shows well-known port numbers for components in a PCMM environment:

  - RADIUS server—1812
  - RADIUS accounting—1813
  - COPS-PR (used for communication between the SAE and CMTS device)—3918

- Property name—targetPorts

Configuring IPSec Properties to Establish Key Exchange and SAs

Use the IPSec Details pane to configure properties to establish IKE, also referred to a phase 1 IKE exchange, and to set up an SA between peers, also referred to as phase 2 exchange. SDX Configuration Editor supplies default values for all fields. You can change values as needed.



IKE Authentication Method

- Authentication method used for IKE.
- Value—preshared key

NOTE: This value cannot be changed.

- Guidelines—This is a required entry.
- Property name—ikeAuthMethod

IKE Encryption Algorithm

- Encryption algorithm for use by during IKE negotiation.
- Values
  - DES
  - 3DES
  - AES
  - Blowfish
- Guidelines—This is a required entry.
- Default—DES
- Property name—ikeEncAlg

IKE Authentication Algorithm

- Authentication algorithm for use during IKE negotiation.
- Values
  - HMAC-MD5
  - HMAC-SHA-1
- Guidelines—This is a required entry.
- Default—HMAC-SHA-1
- Property name—ikeAuthAlg

IKE Oakley Group

- An Oakley group, the type of Diffie-Hellman key exchange algorithm that the Oakley key exchange protocol uses to distribute keying information during IKE negotiation. The Diffie-Hellman key exchange algorithm provides a way for two parties to exchange keying information and to agree on a shared key.
- Value
  - 1—768-bit Diffie-Hellman group
  - 2—1,024-bit Diffie-Hellman group
  - 5—1,536-bit Diffie-Hellman group
- Guidelines—This is a required entry.

  Group 1 provides the weakest security and group 5 the strongest security.
- Default—5
- Property name—ikeOakleyGroup

IKE Lifetime

- Length of time phase 1 SA can be active.
- Value—Length of time in seconds
- Default—28800
- Property name—ikeLifetime

Phase 2 Encryption Algorithm

- Encryption algorithm for use by IKE and is used during negotiation of the security association between hosts.
- Values
  - DES
  - 3DES
  - AES
  - Blowfish
- Guidelines—This is a required entry.

- Default—DES
- Property name—phase2EncAlg

Phase 2 Authentication Algorithm

- Authentication algorithm for use by IKE during negotiation of the security association between hosts.
- Value
  - HMAC-MD5
  - HMAC-SHA-1
- Guidelines—This is a required entry.
- Default—HMAC-SHA1
- Property name—phase2AuthAlg

Phase 2 Oakley Group

- An Oakley group, the type of Diffie-Hellman key exchange algorithm that the Oakley key exchange protocol uses to distribute keying information during SA negotiation. The Diffie-Hellman key exchange algorithm provides a way for two parties to exchange keying information and to agree on a shared key.
- Value
  - 1—768-bit Diffie-Hellman group
  - 2—1,024-bit Diffie-Hellman group
  - 5—1536-bit Diffie-Hellman group
- Guidelines—This is a required entry.

  Group 1 provides the weakest security and group 5 the strongest security.
- Default—5
- Property name—phase2OakleyGroup

Phase 2 Lifetime

- How long the SA between hosts can be active. At the end of the interval specified, the system refreshes the encryption key.
- Value— Length of time
- Default—28800 seconds
- Property name—phase2Lifetime

Applying the IPSec Configuration

After you configure IPSec properties, you can export the configuration properties to the Solaris operating system. The properties are applied to IPSec configuration for the Solaris platform on which the SAE is running.

To apply IPSec configuration properties.

1. In the navigation pane of SDX Configuration Editor, right-click the IPSec object, select **SDX System Configuration**, and then select **Export IPSec to Host**.

2. Select the host to which to export the configuration, and provide a password if you are using SSH between hosts.

   The Solaris platform activates the IPSec configuration.

Changing IPSec Configuration

To configure IPSec attributes from SDX Configuration Editor:

1. In the navigation pane of SDX Configuration Editor, double-click an IPSec object.

2. In the IPSec Transport Connections pane, change field values.

3. In the navigation pane, right-click the IPSec object, select **SDX System Configuration**, and then select **Export IPSec to Host**.

   The Solaris platform activates the updated IPSec configuration.

4. Make corresponding configuration changes on the system with which the SAE has IPSec-protected communication.

5. Test the updated configuration.

Configuring IPSec on a Remote System

For another system, such as a RADIUS server or a CMTS device, and the SAE to participate in IPSec-protected communications, make sure that the IPSec configuration for the remote system includes the values in Table 8. The table describes configuration properties as phase 1 or phase 2. Phase 1 indicates IKE phase 1 exchange and phase 2 indicates IKE phase 2 exchange.

Table 8: Configuration Properties for Remote Hosts

Configuration Property	Description of Value
IKE Configuration	
Phase 1 local identity type	IPv4
Phase 1 remote identity type	IPv4
IKE local identity	IP address for the application (CMTS device or RADIUS)
IKE remote identity	IP address of the SAE

Table 8: Configuration Properties for Remote Hosts (continued)

Configuration Property	Description of Value
Phase 1 authentication method	Preshared key
Phase 1 encryption algorithm	IKE encryption algorithm configured on the SAE
Phase 1 authentication algorithm	IKE authentication algorithm configured on the SAE
Phase 1 IKE mode	Main mode
Phase 1 Perfect Forward Security (PFS) group	IKE Oakley group configured on the SAE
Phase 1 lifetime	IKE lifetime configured on the SAE
Preshared key	Preshared key configured for the SAE
IPSec policy to secure traffic flow	Policy that ensures that traffic between applications is protected; for example, between SAE and RADIUS, or between SAE and CMTS device over COPS-PR
IPSec Policy Configuration	
Phase 2 encryption algorithm	Value configured on the SAE
Phase 2 authentication algorithm	Value configured on the SAE
Phase 2 PFS group	Phase 2 Oakley group configured on the SAE
Phase 2 lifetime	Value configured on the SAE

Testing the IPSec Connection

After you configure IPSec on the system running the SAE and on a remote host, make sure that the hosts are communicating over the connection. For information about testing and troubleshooting IPSec connections, see the IPSec documentation for the system running the SAE and the documentation for the remote system.

Chapter 11

Using PCMM Policy Servers

This chapter describes the Juniper Policy Server (JPS), a component of the SRC software that acts as a policy server in the PacketCable Multimedia Specification (PCMM) environment. It contains the following topics:

- Overview of the JPS on page 105
- JPS Framework on page 106
- JPS Interfaces on page 107
- Before You Configure the JPS on page 107

Overview of the JPS

In a PCMM environment, the policy server acts as a policy decision point (PDP) and policy enforcement point (PEP) that manages the relationships between application managers and cable management termination system (CMTS) devices.

The JPS is a PCMM-compliant policy server. The JPS must be deployed in an SRC environment that satisfies these conditions:

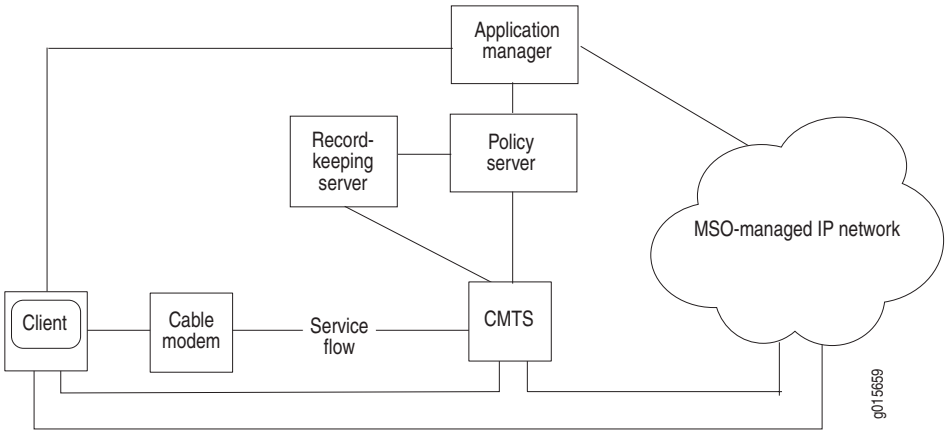
- Organizes PCMM devices into groups (for example, one or more per POP). For redundancy, a community of two or more JPSs will manage each group of PCMM devices.
- Achieves successful state synchronization by requiring an application manager (for example, a pair of redundant SAEs) to talk to one JPS instance at a time.
- Uses IPSec connections for the network interfaces.

For more information about PCMM and the SRC software, see *Chapter 4, Providing Premium Services in a PCMM Environment*.

JPS Framework

Figure 18 depicts the PCMM architectural framework. The JPS communicates with application managers, CMTS devices, and record-keeping servers.

Figure 18: PCMM Architectural Framework



The interactions between the various PCMM components are centered on the policy server. In the PCMM architecture, these basic interactions occur:

1. A client requests a multimedia service from an application manager.
2. Depending on the client type and its QoS signaling capabilities, the application manager relays the request to a policy server.
3. The policy server relays the request to the CMTS device and is responsible for provisioning the policies on a CMTS device.

   Depending on the request, the policy server records an event for the policy request and provides that information to the record-keeping server (RKS).
4. The CMTS device performs admission control and manages network resources through Data over Cable Service Interface Specifications (DOCSIS) service flows based on the provisioned policies.
5. The RKS receives event messages from other network elements, such as the policy server or CMTS device, and acts as a short-term repository for the messages.

JPS Interfaces

The JPS has interfaces, implemented as plug-ins, to communicate with:

- Application managers, such as the SAE
- Record-keeping servers
- CMTS devices

The JPS is relatively stateless, but the individual plug-ins can be stateful.

The JPS uses the Common Open Policy Service (COPS) protocol as specified in the PacketCable Multimedia Specification PKT-SP-MM-I03-051221 for its interface connections. The JPS communicates with the CMTS device and the application manager by using a COPS over Transmission Control Protocol (TCP) connection.

Application Manager to Policy Server Interface

To allow the JPS to communicate with the application manager, this interface accepts and manages COPS over TCP connections from application managers, such as the SAE.

Policy Server to RKS Interface

To allow the JPS to communicate with a set of redundant record-keeping servers, this interface sends a policy event message to the RKS when receiving a PCMM-COPS gate control (request, delete, update) message. This interface also sends time change events to the RKS.

Policy Server to CMTS Interface

To allow the JPS to communicate with policy enforcement points (PCMM devices), this interface initiates and manages COPS over TCP connections with CMTS devices.

Before You Configure the JPS

Before you configure the JPS, deploy an SRC-managed PCMM network. For more information about PCMM and the SRC software, see *Chapter 4, Providing Premium Services in a PCMM Environment*.

You can configure the JPS on a Solaris platform or on a C-series platform.

- To configure the JPS on a C-series platform, see *Chapter 12, Configuring the JPS with the SRC CLI*.
- To use the JPS on a Solaris platform, see *Chapter 13, Configuring the JPS on a Solaris Platform*.

Chapter 12

Configuring the JPS with the SRC CLI

This chapter describes how to use the SRC CLI to configure the Juniper Policy Server (JPS), a component of the SRC software that acts as a policy server in the PacketCable Multimedia Specification (PCMM) environment. You can use the CLI to configure the JPS on a Solaris platform or on a C-series platform.

Topics include:

- Configuration Statements for the JPS on page 109
- Configuring the JPS on page 111
- Modifying the JPS Configuration on page 112
- Modifying the Subscriber Configuration on page 124
- Configuring the SAE to Interact with the JPS on page 125
- Using the NIC Resolver on page 130
- Managing the JPS on page 131

For information about the JPS, see *Chapter 11, Using PCMM Policy Servers*.

Configuration Statements for the JPS

Use the following configuration statements to configure the JPS at the [edit] hierarchy level.

```
slot number jps {
    java-heap-size java-heap-size;
    snmp-agent;
    policy-server-id policy-server-id;
    use-psid-in-gate-commands;
    cmts-message-buffer-size cmts-message-buffer-size;
    am-message-buffer-size am-message-buffer-size;
}

slot number jps am-interface {
    pep-id pep-id;
    listening-address listening-address;
    validate-pcmm-objects;
    message-max-length message-max-length;
```

```
        message-read-buffer-size message-read-buffer-size;
        message-write-buffer-size message-write-buffer-size;
        open-connection-timeout open-connection-timeout;
}

slot number jps cmts-interface {
        cmts-addresses [cmts-addresses...];
        keepalive-interval keepalive-interval;
        synch-despite-unreachable-pep;
        synch-despite-pre-i03-pep;
        use-ssq-ssc-with-pre-i03-pep;
        local-address local-address;
        message-max-length message-max-length;
        message-read-buffer-size message-read-buffer-size;
        message-write-buffer-size message-write-buffer-size;
        open-connection-timeout open-connection-timeout;
        connection-open-retry-interval connection-open-retry-interval;
        sent-message-timeout sent-message-timeout;
        validate-pcmm-objects;
}

slot number jps cmts-registry cmts cmts-ip ...

slot number jps cmts-registry cmts cmts-ip range-pool pool-index {
        low low;
        high high;
}

slot number jps cmts-registry cmts cmts-ip subnet-pool subnet {
        exclude [exclude];
}

slot number jps logger name ...

slot number jps logger name file {
        filter filter;
        filename filename;
        rollover-filename rollover-filename;
        maximum-file-size maximum-file-size;
}

slot number jps logger name syslog {
        filter filter;
        host host;
        facility facility;
        format format;
}

slot number jps rks-interface {
        element-id element-id;
        local-address local-address;
        local-port local-port;
        retry-interval retry-interval;
        local-timeout local-timeout;
        mso-data mso-data;
        mso-domain-name mso-domain-name;
        default-rks-pair default-rks-pair;
```

```
        pending-rks-event-max-size pending-rks-event-max-size;
        pending-rks-event-max-age pending-rks-event-max-age;
        held-decs-max-size held-decs-max-size;
        held-decs-max-age held-decs-max-age;
        bcid-cache-size bcid-cache-size;
        bcid-cache-age bcid-cache-age;
        use-default-when-am-requests-unconfigured-rks;
}

slot number jps rks-interface am am-name {
        am-id am-id;
        rks-pair-name rks-pair-name;
        trusted;
}

slot number jps rks-interface rks-pair rks-pair-name {
        primary-address primary-address;
        primary-port primary-port;
        secondary-address secondary-address;
        secondary-port secondary-port;
}
```

For detailed information about each configuration statement, see the *SRC-PE CLI Command Reference*.

Configuring the JPS

You can modify the JPS configuration, which includes configuring the logging destinations and connections to the JPS interfaces. Any configuration changes will be applied within 15 seconds.

You can configure the subscriber configuration, which maps a subscriber address to the CMTS address.

The tasks to configure the JPS for a cable network environment are:

1. Modifying the JPS Configuration on page 112
2. Modifying the Subscriber Configuration on page 124

In addition to configuring the JPS, you might need to perform these tasks:

1. Configuring the SAE to Interact with the JPS on page 125
2. Using the NIC Resolver on page 130

Modifying the JPS Configuration

To modify the current JPS configuration:

1. Configure general properties for the JPS, including Java heap memory, maximum number of buffered messages for CMTS and application manager destinations, and policy server identifiers.

   See *Configuring General Properties for the JPS* on page 112.

   See *Specifying Policy Server Identifiers in Messages* on page 113.

2. Configure logging destinations for the JPS.

   See *Configuring Logging Destinations for the JPS* on page 114.

3. Configure the connections to the JPS interfaces.

   See *Specifying Connections to the Application Managers* on page 115.

   See *Specifying Connections to RKSs* on page 116.

   See *Specifying Connections to CMTS Devices* on page 121.

Configuring General Properties for the JPS

Use the following configuration statements to configure general properties for the JPS:

```
slot number jps {
    java-heap-size java-heap-size;
    snmp-agent;
    cmts-message-buffer-size cmts-message-buffer-size;
    am-message-buffer-size am-message-buffer-size;
}
```

To configure general properties for the JPS:

1. From configuration mode, access the configuration statement that configures the general properties.

   ```
   user@host# edit slot 0 jps
   ```

2. (Optional) Specify the maximum amount of memory available to the JRE.

   ```
   [edit slot 0 jps]
   user@host# set java-heap-size java-heap-size
   ```

3. (Optional) Enable the JPS to communicate with the SNMP agent.

   ```
   [edit slot 0 jps]
   user@host# set snmp-agent
   ```

4. (Optional) Specify the maximum number of messages buffered for each CMTS destination.

   [edit slot 0 jps]
   user@host# **set cmts-message-buffer-size** *cmts-message-buffer-size*

5. (Optional) Specify the maximum number of messages buffered for each application manager destination.

   [edit slot 0 jps]
   user@host# **set am-message-buffer-size** *am-message-buffer-size*

6. (Optional) Verify your configuration.

   ```
   [edit slot 0 jps]
   user@host# show
   ```

Specifying Policy Server Identifiers in Messages

Use the following configuration statements to configure policy server identifiers for the JPS:

slot *number* jps {
    policy-server-id *policy-server-id*;
    use-psid-in-gate-commands;
}

To configure policy server identifiers for the JPS:

1. From configuration mode, access the configuration statement that configures the policy server identifiers.

   user@host# **edit slot 0 jps**

2. (Optional) Specify the policy server identifier so that the JPS can be identified in messages sent to CMTS devices.

   [edit slot 0 jps]
   user@host# **set policy-server-id** *policy-server-id*

3. (Optional) Configure the JPS so that the policy server identifier is specified in messages sent to the RKS.

   [edit slot 0 jps]
   user@host# **set use-psid-in-gate-commands**

   When the JPS is communicating only with PCMM I03 CMTS devices, the value must be true. When the JPS is communicating with any pre-PCMM I03 CMTS devices, the value must be false.

4. (Optional) Verify your configuration.

   ```
   [edit slot 0 jps]
   user@host# show
   ```

Configuring Logging Destinations for the JPS

By default, the JPS has four logging destinations.

Use the following configuration statements to configure logging destinations for the JPS:

slot *number* jps logger *name* ...

slot *number* jps logger *name* file {
    filter *filter*;
    filename *filename*;
    rollover-filename *rollover-filename*;
    maximum-file-size *maximum-file-size*;
}

slot *number* jps logger *name* syslog {
    filter *filter*;
    host *host*;
    facility *facility*;
    format *format*;
}

Configuring Logging Destinations to Store Messages in a File

To configure logging destinations to store log messages in a file:

1. From configuration mode, access the configuration statement that configures the name and type of logging destination. In this sample procedure, the logging destination called log2 is configured.

   user@host# **edit slot 0 jps logger log2 file**

2. Specify the properties for the logging destination.

   [edit slot 0 jps logger log2 file]
   user@host# **set** ?

   For more information about configuring properties for the logging destination, see *SDX Monitoring and Troubleshooting Guide, Chapter 3, Configuring Logging for SRC Components with the CLI*.

3. (Optional) Verify your configuration.

```
[edit slot 0 jps logger log2]
user@host# show
file {
  filter !NoAckRksEvent,/info-;
  filename var/log/jps_info.log;
  rollover-filename var/log/jps_info.alt;
  maximum-file-size 2000000000;
}
```

Configuring Logging Destinations to Send Messages to System Logging Facility

To configure logging destinations to send log messages to the system logging facility:

1. From configuration mode, access the configuration statement that configures the name and type of logging destination. In this sample procedure, the logging destination called log5 is configured.

   user@host# **edit slot 0 jps logger log5 syslog**

2. Specify the properties for the logging destination.

   [edit slot 0 jps logger log5 syslog]
   user@host# **set** ?

   For more information about configuring properties for the logging destination, see *SDX Monitoring and Troubleshooting Guide, Chapter 3, Configuring Logging for SRC Components with the CLI*.

3. (Optional) Verify your configuration.

   ```
   [edit slot 0 jps logger log5]
   user@host# show
   ```

Specifying Connections to the Application Managers

Use the following configuration statement to configure the application manager–to–policy server interface (PKT-MM3) so that the policy server can communicate with application managers:

slot *number* jps am-interface {
    pep-id *pep-id*;
    listening-address *listening-address*;
    validate-pcmm-objects;
    message-max-length *message-max-length*;
    message-read-buffer-size *message-read-buffer-size*;
    message-write-buffer-size *message-write-buffer-size*;
    open-connection-timeout *open-connection-timeout*;
}

To configure the connections to the application managers:

1. From configuration mode, access the configuration statement that configures the application manager–to–policy server interface.

   user@host# **edit slot 0 jps am-interface**

2. (Optional) Specify the network-wide unique identifier for this JPS instance.

   [edit slot 0 jps am-interface]
   user@host# **set pep-id** *pep-id*

   Changes apply only to COPS connections that are established after you make the change.

3. (Optional) Specify the local IP address on which the JPS listens for incoming connections from application managers.

   [edit slot 0 jps am-interface]
   user@host# **set listening-address** *listening-address*

   Changes take effect only after you restart the JPS (see *Restarting the JPS* on page 131).

4. (Optional) Specify whether to validate PCMM objects received from PDPs.

   [edit slot 0 jps am-interface]
   user@host# **set validate-pcmm-objects**

5. (Optional) Specify the maximum length of incoming messages.

   [edit slot 0 jps am-interface]
   user@host# **set message-max-length** *message-max-length*

6. (Optional) Specify the size of message read buffer.

   [edit slot 0 jps am-interface]
   user@host# **set message-read-buffer-size** *message-read-buffer-size*

7. (Optional) Specify the size of message write buffer.

   [edit slot 0 jps am-interface]
   user@host# **set message-write-buffer-size** *message-write-buffer-size*

8. (Optional) Specify the maximum time to wait for the initial PCMM messages to be exchanged after a TCP connection is established.

   [edit slot 0 jps am-interface]
   user@host# **set open-connection-timeout** *open-connection-timeout*

   The connection is dropped when initial PCMM messages are not exchanged within this time period.

9. (Optional) Verify your configuration.

```
[edit slot 0 jps am-interface]
user@host# show
pep-id SDX-JPS;
listening-address ;
validate-pcmm-objects;
message-max-length 204800;
message-read-buffer-size 1000000;
message-write-buffer-size 1000000;
open-connection-timeout 5;
```

Specifying Connections to RKSs

To configure the policy server–to–RKS interface (PKT-MM4) so that policy events can be sent to the RKS, you can configure RKS pairs (see *Configuring RKS Pairs* on page 119) and their associated application managers (see *Configuring RKS Pairs for Associated Application Managers* on page 120).

Use the following configuration statement to configure the policy server–to–RKS interface:

```
slot number jps rks-interface {
    element-id element-id;
    local-address local-address;
    local-port local-port;
    retry-interval retry-interval;
    local-timeout local-timeout;
    mso-data mso-data;
    mso-domain-name mso-domain-name;
    default-rks-pair default-rks-pair;
    pending-rks-event-max-size pending-rks-event-max-size;
    pending-rks-event-max-age pending-rks-event-max-age;
    held-decs-max-size held-decs-max-size;
    held-decs-max-age held-decs-max-age;
    bcid-cache-size bcid-cache-size;
    bcid-cache-age bcid-cache-age;
    use-default-when-am-requests-unconfigured-rks;
}
```

To configure the policy server–to–RKS interface:

1. From configuration mode, access the configuration statement that configures the policy server–to–RKS interface.

   ```
   user@host# edit slot 0 jps rks-interface
   ```

2. Specify the network-wide unique identifier for RKS event origin.

   ```
   [edit slot 0 jps rks-interface]
   user@host# set element-id element-id
   ```

3. (Optional) Specify the source IP address that the plug-in uses to communicate with the RKS.

   ```
   [edit slot 0 jps rks-interface]
   user@host# set local-address local-address
   ```

   If no value is specified and there is more than one local address, the JPS randomly selects a local address to be used as the source address.

4. (Optional) Specify the source UDP port or a pool of ports that the plug-in uses to communicate with the RKS.

   ```
   [edit slot 0 jps rks-interface]
   user@host# set local-port local-port
   ```

5. (Optional) Specify the time the JPS waits for a response from an RKS before it resends the packet.

   ```
   [edit slot 0 jps rks-interface]
   user@host# set retry-interval retry-interval
   ```

   The JPS keeps sending packets until either the RKS acknowledges the packet or the maximum timeout is reached.

6. (Optional) Specify the maximum time the JPS waits for a response from an RKS.

   [edit slot 0 jps rks-interface]
   user@host# **set local-timeout** *local-timeout*

7. (Optional) Specify the MSO-defined data in the financial entity ID (FEID) attribute, which is included in event messages.

   [edit slot 0 jps rks-interface]
   user@host# **set mso-data** *mso-data*

8. (Optional) Specify the MSO domain name in the FEID attribute that uniquely identifies the MSO for billing and settlement purposes.

   [edit slot 0 jps rks-interface]
   user@host# **set mso-domain-name** *mso-domain-name*

9. (Optional) Specify the default RKS pair that the JPS uses unless an RKS pair is configured for a given application manager.

   [edit slot 0 jps rks-interface]
   user@host# **set default-rks-pair** *default-rks-pair*

10. (Optional) Specify the maximum number of RKS events waiting for Gate-Set-Ack, Gate-Set-Err, Gate-Del-Ack, and Gate-Del-Err messages.

    [edit slot 0 jps rks-interface]
    user@host# **set pending-rks-event-max-size** *pending-rks-event-max-size*

11. (Optional) Specify the oldest age of RKS events waiting for Gate-Set-Ack, Gate-Set-Err, Gate-Del-Ack, and Gate-Del-Err messages.

    [edit slot 0 jps rks-interface]
    user@host# **set pending-rks-event-max-age** *pending-rks-event-max-age*

    The maximum age must be greater than sent-message-timeout of the corresponding CMTS interface.

12. (Optional) Specify the maximum number of outstanding Gate-Info requests.

    [edit slot 0 jps rks-interface]
    user@host# **set held-decs-max-size** *held-decs-max-size*

13. (Optional) Specify the oldest age of outstanding Gate-Info requests.

    [edit slot 0 jps rks-interface]
    user@host# **set held-decs-max-age** *held-decs-max-age*

    The maximum age must be greater than sent-message-timeout of the corresponding CMTS interface.

14. (Optional) Specify the size of billing correlation ID (BCID) cache.

    [edit slot 0 jps rks-interface]
    user@host# **set bcid-cache-size** *bcid-cache-size*

15. (Optional) Specify the oldest age of billing correlation ID (BCID) in cache.

    [edit slot 0 jps rks-interface]
    user@host# **set bcid-cache-age** *bcid-cache-age*

16. (Optional) Specify whether the default RKS pair is used when an application manager requests the use of an unconfigured RKS pair.

    [edit slot 0 jps rks-interface]
    user@host# **set use-default-when-am-requests-unconfigured-rks**

17. (Optional) Verify your configuration.

    ```
    [edit slot 0 jps rks-interface]
    user@host# show
    ```

Configuring RKS Pairs

By default, the JPS has four RKS pairs. All parameters that share the same RKS pair name configure the connection to that RKS pair. Any configured RKS pair can be used as the value for the default RKS pair or the RKS pair associated with a specific application manager.

NOTE: When running more than one JPS in a group to provide redundancy, all the JPSs in that group must have same RKS pair configuration (including the default RKS pair and any configured RKS pairs associated with a specific application manager).

Use the following configuration statement to configure the RKS pair:

slot *number* jps rks-interface rks-pair *rks-pair-name* {
    primary-address *primary-address*;
    primary-port *primary-port*;
    secondary-address *secondary-address*;
    secondary-port *secondary-port*;
}

To configure the RKS pair:

1. From configuration mode, access the configuration statement that configures the RKS pair. In this sample procedure, the RKS pair called pair1 is configured.

   user@host# **edit slot 0 jps rks-interface rks-pair pair1**

2. Specify the IP address of the primary RKS for this RKS pair.

   [edit slot 0 jps rks-interface rks-pair pair1]
   user@host# **set primary-address** *primary-address*

   If no value is specified, the RKS pair is not defined.

3. (Optional) Specify the UDP port on the primary RKS to which the JPS sends events.

   [edit slot 0 jps rks-interface rks-pair pair1]
   user@host# **set primary-port** *primary-port*

4. (Optional) Specify the IP address of the secondary RKS for this RKS pair.

   [edit slot 0 jps rks-interface rks-pair pair1]
   user@host# **set secondary-address** *secondary-address*

5. (Optional) Specify the UDP port on the secondary RKS to which the JPS sends events.

   [edit slot 0 jps rks-interface rks-pair pair1]
   user@host# **set secondary-port** *secondary-port*

6. (Optional) Verify your configuration.

```
[edit slot 0 jps rks-interface rks-pair pair1]
user@host# show
primary-address ;
primary-port 1813;
secondary-address ;
secondary-port 1813;
```

Configuring RKS Pairs for Associated Application Managers

By default, the JPS has four associated application managers. All parameters that share the same application manager name configure the RKS pair to which events associated with a specific application manager are sent.

Use the following configuration statement to configure the associated application manager:

slot *number* jps rks-interface am *am-name* {
    am-id *am-id*;
    rks-pair-name *rks-pair-name*;
    trusted;
}

To configure the associated application manager:

1. From configuration mode, access the configuration statement that configures the RKS pair for the associated application manager. In this sample procedure, the application manager name called 1 is configured.

   user@host# **edit slot 0 jps rks-interface am 1**

2. Specify the identifier of the application manager.

   [edit slot 0 jps rks-interface am 1]
   user@host# **set am-id** *am-id*

If no value is specified, the RKS pair configuration is not defined for this application manager. If you must set trusted to true without defining the RKS pair configuration, you must specify a value for am-id and not specify a value for rks-pair-name.

3. (Optional) Specify the RKS pair that the JPS will send events to when those events are triggered by gate transitions associated with the application manager specified by am-id with the same application manager name (am-name).

   [edit slot 0 jps rks-interface am 1]
   user@host# **set rks-pair** *rks-pair-name*

   If no value is specified, the RKS pair configuration is not defined for this application manager. Use when you must set trusted to true without defining the RKS pair configuration.

4. (Optional) Specify whether this application manager is a trusted network element to the JPS.

   [edit slot 0 jps rks-interface am 1]
   user@host# **set trusted**

5. (Optional) Verify your configuration.

   ```
   [edit slot 0 jps rks-interface am 1]
   user@host# show
   ```

Specifying Connections to CMTS Devices

Use the following configuration statement to configure the policy server–to–CMTS interface (PKT-MM2) so that the policy server can communicate with CMTS devices:

slot *number* jps cmts-interface {
    cmts-addresses [*cmts-addresses*...];
    keepalive-interval *keepalive-interval*;
    synch-despite-unreachable-pep;
    synch-despite-pre-i03-pep;
    use-ssq-ssc-with-pre-i03-pep;
    local-address *local-address*;
    message-max-length *message-max-length*;
    message-read-buffer-size *message-read-buffer-size*;
    message-write-buffer-size *message-write-buffer-size*;
    open-connection-timeout *open-connection-timeout*;
    connection-open-retry-interval *connection-open-retry-interval*;
    sent-message-timeout *sent-message-timeout*;
    validate-pcmm-objects;
}

To configure the policy server–to–CMTS interface:

1. From configuration mode, access the configuration statement that configures the policy server–to–CMTS interface.

   user@host# **edit slot 0 jps cmts-interface**

2. Specify the IP addresses of all the CMTS devices to which the JPS will try to connect.

   [edit slot 0 jps cmts-interface]
   user@host# **set cmts-addresses** [*cmts-addresses*...]

3. (Optional) Specify the interval between keepalive messages sent from the COPS client (CMTS device) to the COPS server (the JPS). Changes apply only to COPS connections that are established after you make the change.

   [edit slot 0 jps cmts-interface]
   user@host# **set keepalive-interval** *keepalive-interval*

   A value of 0 means that no keepalive messages will be exchanged between the CMTS device and the JPS.

4. (Optional) Specify whether synchronization proceeds when the JPS receives a synchronization request from an application manager (such as the SAE) and the JPS is not connected to a CMTS device to which it should be connected.

   [edit slot 0 jps cmts-interface]
   user@host# **set synch-despite-unreachable-pep**

5. (Optional) Specify whether synchronization proceeds when the JPS receives a synchronization request from an application manager (such as the SAE) and the JPS is connected to a pre-PCMM I03 CMTS device.

   [edit slot 0 jps cmts-interface]
   user@host# **set synch-despite-pre-i03-pep**

6. (Optional) Specify whether synchronization includes both pre-PCMM I03 and PCMM I03 CMTS devices when the JPS receives a synchronization request from an application manager (such as the SAE) and the JPS is connected to a pre-PCMM I03 CMTS device. Relevant only when at least one pre-PCMM I03 CMTS device is connected and sync-despite-pre-i03-pep is specified as true.

   [edit slot 0 jps cmts-interface]
   user@host# **set use-ssq-ssc-with-pre-i03-pep**

7. (Optional) Specify the source IP address that the JPS uses to communicate with CMTS devices.

   [edit slot 0 jps cmts-interface]
   user@host# **set local-address** *local-address*

   If no value is specified and there is more than one local address, a random local address is used as the source address.

8. (Optional) Specify the maximum length of incoming messages.

   [edit slot 0 jps cmts-interface]
   user@host# **set message-max-length** *message-max-length*

9. (Optional) Specify the size of message read buffer.

   [edit slot 0 jps cmts-interface]
   user@host# **set message-read-buffer-size** *message-read-buffer-size*

10. (Optional) Specify the size of message write buffer.

    [edit slot 0 jps cmts-interface]
    user@host# **set message-write-buffer-size** *message-write-buffer-size*

11. (Optional) Specify the maximum time to wait for the initial PCMM messages to be exchanged after a TCP connection is established.

    [edit slot 0 jps cmts-interface]
    user@host# **set open-connection-timeout** *open-connection-timeout*

    The connection is dropped when initial PCMM messages are not exchanged within this time period.

12. (Optional) Specify the time to wait before the JPS tries to reconnect to CMTS devices.

    [edit slot 0 jps cmts-interface]
    user@host# **set connection-open-retry-interval** *connection-open-retry-interval*

13. (Optional) Specify the maximum time to wait for the sent messages to be exchanged after a TCP connection is established.

    [edit slot 0 jps cmts-interface]
    user@host# **set sent-message-timeout** *sent-message-timeout*

    This value must be less than the held-decs-max-age and pending-rks-event-max-age values for the corresponding RKS interface.

14. (Optional) Specify whether to validate PCMM objects received from PDPs.

    [edit slot 0 jps cmts-interface]
    user@host# **set validate-pcmm-objects**

15. (Optional) Verify your configuration.

```
[edit slot 0 jps cmts-interface]
user@host# show
cmts-addresses ;
keepalive-interval 60;
synch-despite-unreachable-pep;
synch-despite-pre-i03-pep;
local-address ;
message-max-length 204800;
message-read-buffer-size 1000000;
message-write-buffer-size 1000000;
open-connection-timeout 5;
```

```
connection-open-retry-interval 60;
sent-message-timeout 60;
validate-pcmm-objects;
```

Modifying the Subscriber Configuration

To locate the CMTS device associated with a subscriber, the JPS maps the subscriber IP address in a message to the CMTS IP address to which the message must be delivered. This mapping specifies the subscriber IP pools associated with CMTS devices.

Use the following configuration statements to configure a CMTS device to which the JPS can connect and the pools of subscriber IP addresses that are managed by the CMTS device:

slot *number* jps cmts-registry cmts *cmts-ip* ...

slot *number* jps cmts-registry cmts *cmts-ip* range-pool *pool-index* {
    low *low*;
    high *high*;
}

slot *number* jps cmts-registry cmts *cmts-ip* subnet-pool *subnet* {
    exclude [*exclude*];
}

Configuring Subscriber IP Pools as IP Address Ranges

To configure subscriber IP pools that are managed by the CMTS device as IP address ranges:

1. From configuration mode, access the configuration statement that configures the CMTS device to which the JPS can connect.

   user@host# **edit slot 0 jps cmts-registry cmts** *cmts-ip* **range-pool** *pool-index*

   Specify the IP address of the CMTS device and the address range pool index.

2. Specify the first IP address in the IP range for the pool of subscriber IP addresses that are managed by the CMTS device.

   [edit slot 0 jps cmts-registry cmts *cmts-ip* range-pool *pool-index*]
   user@host# **set low** *low*

3. Specify the last IP address in the IP range for the pool of subscriber IP addresses that are managed by the CMTS device.

   [edit slot 0 jps cmts-registry cmts *cmts-ip* range-pool *pool-index*]
   user@host# **set high** *high*

4. (Optional) Verify your configuration.

   ```
   [edit slot 0 jps cmts-registry]
   user@host# show
   ```

Configuring Subscriber IP Pools as IP Subnets

To configure subscriber IP pools that are managed by the CMTS device as IP subnets:

1. From configuration mode, access the configuration statement that configures the CMTS device to which the JPS can connect.

   user@host# **edit slot 0 jps cmts-registry cmts** *cmts-ip* **subnet-pool** *subnet*

   Specify the IP address of the CMTS device and the IP address and mask of the subnet for the pool of subscriber IP addresses.

2. (Optional) Specify the IP addresses of the subnet that are excluded from the subscriber IP pool managed by the CMTS device.

   [edit slot 0 jps cmts-registry cmts *cmts-ip* subnet-pool *subnet*]
   user@host# **set exclude** [*exclude*...]

3. (Optional) Verify your configuration.

   ```
   [edit slot 0 jps cmts-registry]
   user@host# show
   ```

Configuring the SAE to Interact with the JPS

You must configure the SAE as an application manager to allow it to interact with PCMM-compliant policy servers. The policy server acts as a policy decision point that manages the relationships between application managers and CMTS devices. Policy servers that manage the same group of CMTS devices are grouped together and are simultaneously active. The policy server group provides a way for the SAE to communicate with any CMTS device that is managed by a policy server in the policy server group. To provide redundancy, the SAEs are grouped in an SAE community that connects to a policy server group. Only one of the SAEs in the SAE community is active. The active SAE establishes connections to all the policy servers in the policy server group. The active SAE will fail over to a redundant SAE only when it loses the connection to all the policy servers in the policy server group. State synchronization enables the SAE to synchronize its state with all the CMTS devices connected to a policy server group.

The tasks to configure the SAE as an application manager are:

- Specifying Application Managers for the Policy Server on page 126
- Specifying Application Manager Identifiers for Policy Servers on page 127
- Adding Objects for Policy Servers to the Directory on page 128
- Configuring Initialization Scripts on page 128
- Enabling State Synchronization on page 129

Specifying Application Managers for the Policy Server

To specify the SAE community that connects to a policy server group, you need to add an application manager group object to the directory.

Use the following configuration statements to specify the application manager for the policy server:

```
shared network application-manager-group name {
    description description;
    application-manager-id application-manager-id;
    connected-sae [connected-sae...];
    pdp-group pdp-group;
    local-address-pools [local-address-pools...];
    managing-sae-ior managing-sae-ior;
}
```

To add an application manager group:

1. From configuration mode, access the configuration statement that specifies the application managers.

   user@host# **edit shared network application-manager-group** *name*

2. (Optional) Specify information about the SAE community.

   [edit shared network application-manager-group *name*]
   user@host# **set description** *description*

3. (Optional) Specify the unique identifier within the domain of the service provider for the application manager that handles the service session (Application Manager Tag) as a 2-byte unsigned integer.

   [edit shared network application-manager-group *name*]
   user@host# **set application-manager-id** *application-manager-id*

4. (Optional) Specify the SAEs that are connected to the specified policy server group. This list becomes the community of SAEs.

   [edit shared network application-manager-group *name*]
   user@host# **set connected-sae** [*connected-sae*...]

When you modify a community, wait for passive session stores of the new community members to be updated before you shut down the current active SAE. Otherwise, a failover from the current active SAE to the new member is triggered immediately, and the new member's session store may not have received all data from the active SAE's session store.

5. (Optional) Specify the name of the policy server group associated with this SAE community.

   [edit shared network application-manager-group *name*]
   user@host# **set pdp-group** *pdp-group*

6. (Optional) Specify the list of IP address pools that the specified PDP group currently manages and stores.

   [edit shared network application-manager-group *name*]
   user@host# **set local-address-pools** *local-address-pools*

   You must configure a local address pool if you are using the NIC so that the NIC can resolve the IP-to-SAE mapping. See *Using the NIC Resolver* on page 130.

7. (Optional) Specify the Common Object Request Broker Architecture (CORBA) reference for the SAE managing this policy server group.

   [edit shared network application-manager-group *name*]
   user@host# **set managing-sae-ior** *managing-sae-ior*

   The **amIorPublisher** script provides this information when the SAE connects to the policy server. If you do not select this script when configuring initialization scripts, enter a value. For information about configuring initialization scripts, see *Configuring Initialization Scripts* on page 128.

Specifying Application Manager Identifiers for Policy Servers

The application manager identifier (AMID) identifies the application manager (such as the SAE) in messages sent to and from the policy server. The SAE constructs the AMID value by concatenating two fields: Application Manager Tag and Application Type.

The Application Manager Tag value is obtained from the specification of application managers for policy servers. See *Specifying Application Managers for the Policy Server* on page 126.

The Application Type value is obtained during service activation from the specification of the PCMM Application Type value when you configure normal services. For more information about configuring services, see *SDX Services and Policies Guide, Chapter 1, Managing Services with the SRC CLI*.

Adding Objects for Policy Servers to the Directory

To communicate with policy servers, the SAE creates and manages pseudointerfaces that it associates with a policy decision point object in the directory. Each policy server in the SRC network must appear in the directory as a policy decision point object.

Use the following configuration statements to specify the policy server as a policy decision point:

```
shared network policy-decision-point name {
    description description;
    pdp-address pdp-address;
    pdp-group pdp-group;
}
```

To add a policy server to the directory with the SRC CLI:

1. From configuration mode, access the configuration statement that configures the policy decision point.

   ```
   user@host# edit shared network policy-decision-point name
   ```

2. (Optional) Specify information about the policy server.

   ```
   [edit shared network policy-decision-point name]
   user@host# set description description
   ```

3. (Optional) Specify the IP address of the policy server. The SAE uses this address to establish a COPS connection with the policy server.

   ```
   [edit shared network policy-decision-point name]
   user@host# set pdp-address pdp-address
   ```

4. (Optional) Specify the name of the policy server group.

   ```
   [edit shared network policy-decision-point name]
   user@host# set pdp-group pdp-group
   ```

5. Create an SAE community for the policy servers. See *Specifying Application Managers for the Policy Server* on page 126.

Configuring Initialization Scripts

When the SAE establishes a connection with a policy server, it runs an initialization script to customize the setup of the connection.

Use the following configuration statement to configure the initialization script:

```
shared sae configuration driver scripts {
    pcmm pcmm;
}
```

To configure initialization scripts for the SAE:

1. From configuration mode, access the configuration statement that configures the initialization scripts.

   user@host# **edit shared sae configuration driver scripts**

2. Specify the initialization script for a PCMM environment.

   [edit shared sae configuration driver scripts]
   user@host# **set pcmm** *pcmm*

   The script is run when the connection between a policy server and the SAE is established and again when the connection is dropped. For the JPS, we recommend setting this value to amIorPublisher.

Enabling State Synchronization

State synchronization is achieved when the SAE is required to communicate with the policy server over the COPS connection.

Use the following configuration statement to configure state synchronization:

```
shared sae configuration driver pcmm {
    disable-full-sync;
    disable-pcmm-i03-policy;
    session-recovery-retry-interval session-recovery-retry-interval;
}
```

To enable state synchronization with policy servers:

1. From configuration mode, access the configuration statement that configures the PCMM device driver.

   user@host# **edit shared sae configuration driver pcmm**

2. Specify whether state synchronization with the PCMM policy servers is disabled.

   [edit shared sae configuration driver pcmm]
   user@host# **set disable-full-sync**

   When using other PCMM-compliant policy servers (instead of the JPS), we recommend setting this value to true.

3. Specify whether PCMM I03 policies are disabled when the SAE is deployed with pre-PCMM I03 CMTS devices.

   [edit shared sae configuration driver pcmm]
   user@host# **set disable-pcmm-i03-policy**

   When there are pre-PCMM I03 CMTS devices in the network, you must set this value to true.

4. Specify the time interval between attempts by the SAE to restore service sessions that are still being recovered in the background when state synchronization completes with a state-data-incomplete error.

   [edit shared sae configuration driver pcmm]
   user@host# **set session-recovery-retry-interval** *session-recovery-retry-interval*

   We recommend setting this value to 3600000 (1 hour) or longer.

Using the NIC Resolver

If you are using the NIC to map the subscriber IP address to the SAE, you need to configure a NIC host. The NIC system uses IP address pools to map IP addresses to SAEs. You configure the local address pools in the application manager configuration for a policy server group. These pools are published in the NIC. The NIC maps subscriber IP addresses in requests received through the portal or Advanced Services Gateway to the policy server group that currently manages that CMTS device. For information about configuring the SAE for policy servers, see *Specifying Application Managers for the Policy Server* on page 126.

The OnePopPcmm sample configuration data supports this scenario for a PCMM environment in which you use the assigned IP subscriber method to log in subscribers and in which you use the NIC to determine the subscriber's SAE. The OnePopPcmm configuration supports one point of presence (POP). NIC replication can be used to provide high availability. The realm for this configuration accommodates the situation in which IP pools are configured locally on each application manager group object.

The resolution process takes a subscriber's IP address as the key and returns a reference to the SAE managing this subscriber as the value.

The following agents collect information for resolvers in this realm:

- Directory agent PoolVr collects and publishes information about the mappings of IP pools to the policy server group.
- Directory agent VrSaeId collects and publishes information about the mappings of policy server groups to SAEs.

For more information about configuring the NIC, see *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 10, Configuring NIC with the SRC CLI*.

Managing the JPS

After you have installed the JPS and applied the local configuration of the JPS, you can perform these tasks:

- Starting the JPS on page 131
- Restarting the JPS on page 131
- Stopping the JPS on page 131
- Displaying JPS Status on page 131

To modify the JPS configuration, see *Configuring the JPS* on page 111. To monitor the JPS, see *Chapter 14, Monitoring the JPS with the SRC CLI*.

Starting the JPS

You must start the JPS when you install the JPS without rebooting the JPS host.

To start the JPS:

user@host> **enable component jps**

The system responds with a start message. If the JPS is already running, the system responds with a warning message.

Restarting the JPS

To restart the JPS:

user@host> **restart component jps**

The system responds with a start message. If the JPS is already running, the system responds with a shutdown message and then a start message.

Stopping the JPS

To stop the JPS:

user@host> **disable component jps**

The system responds with a shutdown message. If the JPS is not running when you issue the command, the system responds with the command prompt.

To start the JPS, see *Starting the JPS* on page 131.

Displaying JPS Status

To display the JPS status:

user@host> **show component**

The system responds with a status message.

Chapter 13

Configuring the JPS on a Solaris Platform

This chapter describes how to configure the Juniper Policy Server (JPS), a component of the SRC software that acts as a policy server in the PacketCable Multimedia Specification (PCMM) environment, on a Solaris platform using the SRC configuration applications that run only on Solaris platforms.

You can also use the CLI that runs on Solaris platforms and the C-series platforms to configure the JPS. See *Chapter 12, Configuring the JPS with the SRC CLI*.

This chapter contains the following topics:

- Installing the JPS on page 133
- Starting and Managing the JPS on page 135
- Configuring the JPS on page 137
- Monitoring the JPS on page 145

For more information about the JPS, see *Chapter 11, Using PCMM Policy Servers*.

Installing the JPS

Before you use the JPS for the first time:

1. Deploy an SRC-managed PCMM network.

   For more information about PCMM and the SRC software, see *Chapter 4, Providing Premium Services in a PCMM Environment*.

2. Install the UMCjps package.

   pkgadd -d /cdrom/cdrom0/solaris UMCjps

   For information about installing Solaris packages, see *SDX Getting Started Guide, Chapter 28, Installing the SRC Software on a Solaris Platform*.

3. Apply the local configuration on the host.

   /opt/UMC/jps/etc/config -a

   This command examines the local machine environment, customizes the *etc/jps.in* and *etc/jpsroot.in* files, and installs the customized files as the *etc/jps* and *etc/jpsroot* files.

NOTE: You must apply the local configuration once after installing the JPS.

If you want to configure the JPS to send time change events to the RKS, apply the local configuration using the command described in *Configuring the JPS for Time Change Event Notification* on page 134.

Configuring the JPS for Time Change Event Notification

PCMM-compliant policy servers send time change events to the RKS. You can configure the JPS to send time change events to the RKS by using the Network Time Protocol (NTP) to synchronize time with the local clock.

NOTE: Configuring NTP on the system may interfere with all other time-sensitive components on the system. We recommend that you configure NTP only if the JPS is running on the system by itself.

To configure the JPS to send time change events to the RKS:

1. On the JPS host, log in as `root`.

2. Configure the NTP servers.

   /opt/UMC/jps/etc/config -a -t <ntpServer>,<ntpServer>

   where ntpServer is the DNS name or IP address of an NTP server accessible from the JPS host. Use a comma to separate each NTP server if you specify more than one.

   This command schedules an NTP cron job every 10 minutes to synchronize the local clock for the JPS with the NTP servers. The JPS sends the time change event to the RKS if the local clock changes during synchronization.

Modifying the Local Clock

If you have configured NTP servers for the JPS by using the procedure described in *Configuring the JPS for Time Change Event Notification* on page 134, do not modify the time for the local clock by using the standard **date** command.

To modify the local clock:

1. On the JPS host, log in as `root`.
2. Modify the time for the local clock.

   /opt/UMC/jps/etc/jpsDate [<MMDDhhmm>[[<CC>]<YY>][.<ss>]]

   where MM indicates the month, DD indicates the day, hh indicates the hour, mm indicates the minute, CC indicates the century minus one, YY indicates the last 2 digits of the year, and ss indicates the second.

   The month, day, year, and century may be omitted; the current values are applied as defaults.

   For example, the following entry sets the date to Oct 8, 12:45 AM:

   /opt/UMC/jps/etc/jpsDate 10080045

   The current year is the default because no year is supplied.

Starting and Managing the JPS

After you have installed the JPS and applied the local configuration of the JPS, you can perform these tasks:

- Starting the JPS on page 135
- Restarting the JPS on page 136
- Stopping the JPS on page 136
- Displaying JPS Status on page 136

To modify the JPS configuration, see *Configuring the JPS* on page 137. To monitor the JPS configuration, see *Monitoring the JPS* on page 145.

Starting the JPS

You must start the JPS when you install the JPS without rebooting the JPS host.

To start the JPS:

1. On the JPS host, log in as `root` or as an authorized nonroot admin user.
2. Start the JPS from its installation directory.

   For root user: **/opt/UMC/jps/etc/jps start**

   For nonroot user: **/opt/UMC/jps/etc/jpsroot start**

   The system responds with a start message. If the JPS is already running, the system responds with a warning message.

Restarting the JPS

To restart the JPS:

1. On the JPS host, log in as `root` or as an authorized nonroot admin user.
2. Restart the JPS from its installation directory.

   For root user: **/opt/UMC/jps/etc/jps restart**

   For nonroot user: **/opt/UMC/jps/etc/jpsroot restart**

   The system responds with a start message. If the JPS is already running, the system responds with a shutdown message and then a start message.

Stopping the JPS

To stop the JPS:

1. On the JPS host, log in as `root` or as an authorized nonroot admin user.
2. Stop the JPS from its installation directory.

   For root user: **/opt/UMC/jps/etc/jps stop**

   For nonroot user: **/opt/UMC/jps/etc/jpsroot stop**

   The system responds with a shutdown message. If the JPS is not running when you issue the command, the system responds with the command prompt.

To start the JPS, see *Starting the JPS* on page 135.

Displaying JPS Status

To display the JPS status:

1. On the JPS host, log in as `root` or as an authorized nonroot admin user.
2. Display the status from the JPS installation directory.

   For root user: **/opt/UMC/jps/etc/jps status**

   For nonroot user: **/opt/UMC/jps/etc/jpsroot status**

   The system responds with a status message.

Configuring the JPS

You can configure and manage the JPS by using the SRC CLI that runs on Solaris platforms and the C-series platforms. See *Chapter 12, Configuring the JPS with the SRC CLI*.

The tasks to configure the JPS for a cable network environment are:

1. Configuring the JPS on page 111
2. Modifying the Subscriber Configuration on page 124

In addition to configuring the JPS, you might need to perform these tasks:

1. Configuring the SAE to Interact with the JPS on page 125

   You can also use SRC configuration applications to perform this task. See Configuring the SAE to Interact with the JPS on Solaris Platforms on page 137.

2. Using the NIC Resolver on page 130

Configuring the SAE to Interact with the JPS on Solaris Platforms

You must configure the SAE as an application manager to allow it to interact with PCMM-compliant policy servers. The policy server acts as a policy decision point that manages the relationships between application managers and CMTS devices. Policy servers that manage the same group of CMTS devices are grouped together and are simultaneously active. The policy server group provides a way for the SAE to communicate with any CMTS device that is managed by a policy server in the policy server group. To provide redundancy, the SAEs are grouped in an SAE community that connects to a policy server group. Only one of the SAEs in the SAE community is active. The active SAE establishes connections to all the policy servers in the policy server group. The active SAE will fail over to a redundant SAE only when it loses the connection to all the policy servers in the policy server group. State synchronization enables the SAE to synchronize its state with all the CMTS devices connected to a policy server group.

The tasks to configure the SAE as an application manager are:

- Specifying Application Managers for the Policy Server on page 138
- Specifying Application Manager Identifiers for Policy Servers on page 141
- Adding Objects for Policy Servers to the Directory on page 142
- Configuring Initialization Scripts on page 143
- Enabling State Synchronization on page 144

Specifying Application Managers for the Policy Server

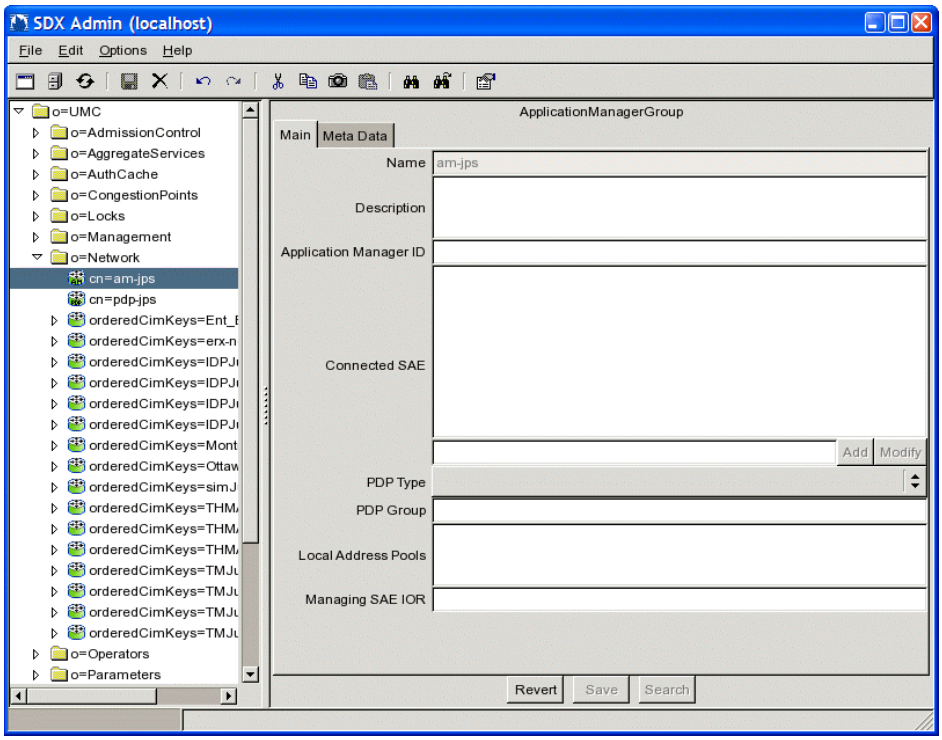
To specify the SAE community that connects to a policy server group, you need to add an application manager group object to the directory.

To add an application manager group with SDX Admin:

1. In the navigation pane, highlight *o = Network*, and right-click.
2. Select **New > ApplicationManagerGroup**.

   The New ApplicationManagerGroup dialog box appears.
3. In the New ApplicationManagerGroup dialog box, enter the name of the application manager group, and click **OK**.

   The name of the group appears in the navigation pane, and information about the group appears in the ApplicationManagerGroup pane.



4. Configure the parameters in the Main tab.
5. Click **Save** in the ApplicationManagerGroup pane.

Description

- Specifies information about the SAE community; keywords that the SDX Admin find utility uses.
- Value—Text string
- Default—No value

Application Manager Tag

- Unique identifier within the domain of the service provider for the application manager that handles the service session; used to specify the application manager identifier (AMID) that is included in all messages sent to and from the policy server.
- Value—2-byte unsigned integer
- Guidelines—This property is required.

  The SAE constructs the AMID value by concatenating two fields: Application Manager Tag and Application Type. The Application Type value is obtained from a service during activation. For more information about the Application Type field, see *Specifying Application Manager Identifiers for Policy Servers* on page 141.

- Default—No value

Connected SAE

- SAEs that are connected to the specified policy server group (PDP Group). This list becomes the community of SAEs.
- Value—IP address or hostname
- Guidelines—This property is required. When you modify a community, wait for passive session stores of the new community members to be updated before you shut down the current active SAE. Otherwise, a failover from the current active SAE to the new member is triggered immediately, and the new member's session store may not have received all data from the active SAE's session store.
- Default—No value

PDP Type

- Type of device that this directory object will be used to manage.
- Value—For the JPS, enter the value PCMM.

  If you do not fill in this field, the device driver ignores this application manager group.

- Default—No value

PDP Group

- Name of the policy server group associated with this SAE community.
- Value—Text string
- Guidelines—This property is required.
- Default—No value

Local Address Pools

- List of IP address pools that the specified PDP group currently manages and stores. You must configure a local address pool if you are using the NIC so that the NIC can resolve the IP-to-SAE mapping. See *Using the NIC Resolver* on page 130.
- Value—List of IP address pools. You can specify an unlimited number of IP address pools. You can specify either the first and last addresses in a range, or you can specify a subnet address, a subnet mask, and a list of addresses to exclude from the subnet.

  The IP pool syntax has the following format:

  ([< ipAddressStart > < ipAddressEnd >] |
  { < ipBaseAddress > /(< mask > | < digitNumber >)(, < ipAddressExclude >)* })

  - < ipAddressStart > —First IP address (version 4 or 6) in a range
  - < ipAddressEnd > —Last IP address (version 4 or 6) in a range
  - < ipBaseAddress > —Network base address
  - < mask > —Subnet mask
  - < digitNumber > —Integer specifying the length of the subnet mask
  - < ipAddressExclude > —List of IP addresses to be excluded from the subnet
  - |—Choice of expression; choose either the expression to the left or the expression to the right of this symbol
  - *—Zero or more instances of the preceding group

  You can use spaces in the syntax only to separate the first and last explicit IP addresses in a range.
- Default—No value
- Example—([10.10.10.5 10.10.10.250] {10.20.20.0/24})

Managing SAE IOR

- Common Object Request Broker Architecture (CORBA) reference for the SAE managing this policy server group.
- Value—One of the following items:
  - The actual CORBA reference for the SAE
  - The absolute path to the interoperable object reference (IOR) file
  - A corbaloc URL in the form corbaloc:: < host > :8801/SAE
    - < host > —Name or IP address of the SAE host

- Guidelines—The **amIorPublisher** script provides this information when the SAE connects to the policy server. If you do not select this script when configuring initialization scripts, enter a value in this field. For information about configuring initialization scripts, see *Configuring Initialization Scripts* on page 143.
- Default—No value
- Example—One of the following items:
  - Absolute path— */opt/UMC/sae/var/run/sae.ior*
  - corbaloc URL—boston:8801/sae
  - Actual IOR—IOR:000000000000002438444C3A736D67742E6A756E697...

Specifying Application Manager Identifiers for Policy Servers

To configure the AMID so that the application manager (such as the SAE) can be identified in messages sent to and from the policy server, the SAE constructs the AMID value by concatenating two fields: Application Manager Tag and Application Type. The Application Manager Tag value is obtained from the specification of application managers for policy servers. The Application Type value is obtained during service activation from the specification of the PCMM Application Type value when you configure normal services. For more information about configuring services, see *SDX Services and Policies Guide, Chapter 2, Managing Services on a Solaris Platform*.

PCMM Application Type

- Unique identifier within the domain of the service provider for the application associated with a gate; used to specify the AMID that is included in all messages sent to and from the policy server.
- Value—2-byte unsigned integer
  - 0—No defined application association
  - Other values—Application Type
- Guidelines—This property is required.

  The SAE constructs the AMID value by concatenating two fields: Application Manager Tag and PCMM Application Type. For more information about the Application Manager Tag field, see *Specifying Application Managers for the Policy Server* on page 138.
- Default—No value

Adding Objects for Policy Servers to the Directory

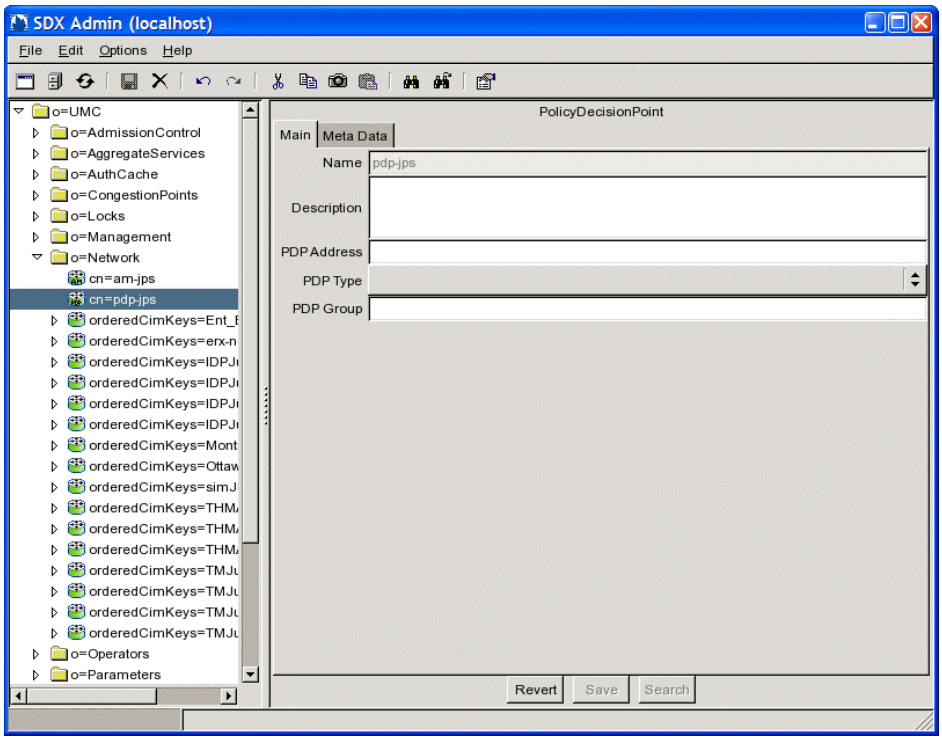
To communicate with policy servers, the SAE creates and manages pseudointerfaces that it associates with a policy decision point object in the directory. Each policy server in the SRC network must appear in the directory as a policy decision point object.

To add a policy server to the directory with SDX Admin:

1. In the navigation pane, select *o = Network*, and right-click.
2. Select **New > PolicyDecisionPoint**.

   The New PolicyDecisionPoint dialog box appears.
3. In the New PolicyDecisionPoint dialog box, enter the name of the policy server, and click **OK**.

   The name of the policy server appears in the navigation pane, and information about the policy server appears in the PolicyDecisionPoint pane.



4. Set the parameters in the Main tab of the PolicyDecisionPoint pane.
5. Click **Save** in the PolicyDecisionPoint pane.
6. Create an SAE community for the policy servers. See *Specifying Application Managers for the Policy Server* on page 138.

Description

- Information about this policy server; keywords that the SDX Admin find utility uses.
- Value—Text string
- Default—No value

PDP Address

- IP address of the policy server. The SAE uses this address to establish a COPS connection with the policy server.
- Value—IP address
- Guidelines—This property is required.
- Default—No value

PDP Type

- Type of device that this directory object will be used to manage.
- Value—For the JPS, enter the value PCMM.

  If you do not fill in this field, the device driver ignores this policy server.
- Default—No value

PDP Group

- Name of the policy server group.
- Value—Text string
- Guidelines—This property is required.
- Default—No value

Configuring Initialization Scripts

When the SAE establishes a connection with a policy server, it runs an initialization script to customize the setup of the connection.

To use SDX Configuration Editor to configure initialization scripts for the SAE:

1. In the navigation pane, select the SAE object for which you want to configure an initialization script.
2. Select the Router tab.

   The Router pane appears.
3. In the Router Scripts area of the Router pane, enter the name of the initialization script in the PCMM Script property.

PCMM Script

- Initialization script for a PCMM environment. The script is run when the connection between a policy server and the SAE is established and again when the connection is dropped.
- Value—Name of a script
- Default—amIorPublisher
- Property name—Router.script.pcmm

Enabling State Synchronization

State synchronization is achieved when the SAE is required to communicate with the policy server over the COPS connection. To enable state synchronization with policy servers, you can specify these properties for the PCMM device driver in the Router tab of SDX Configuration Editor.

Disable Full Sync

- When the SAE is deployed with PCMM policy servers, specifies whether state synchronization with the PCMM policy servers is enabled or disabled.
- Value
  - true—Disables state synchronization
  - false—Enables state synchronization
- Guidelines—When using other PCMM-compliant policy servers (instead of the JPS), we recommend setting this value to true.
- Default—false
- Property name—Router.pcmm.disableStateSync

Disable I03 Policy

- When the SAE is deployed with pre-PCMM I03 CMTS devices, disable the PCMM I03 policies by setting this property to true.
- Value
  - true—Disables PCMM I03-compliant policy
  - false—Enables PCMM I03-compliant policy
- Guidelines—When there are pre-PCMM I03 CMTS devices in the network, you must set this value to true.
- Default—true
- Property name—Router.pcmm.disableI03policy

Session Recovery Retry Interval

- Time interval between attempts by the SAE to restore service sessions that are still being recovered in the background when state synchronization completes with a state-data-incomplete error. The SAE attempts to restore a service session if it receives a service modification or deactivation request for an unrecovered service session before the next interval.
- Value—Number of milliseconds in the range 0–2147483647

- Guidelines—We recommend setting this value to 3600000 (1 hour) or longer.
- Default—3600000
- Property name—Router.pcmm.backgroundSessionRecovery.retryInterval

Monitoring the JPS

You can use the SRC CLI or the C-Web interface to monitor:

- The basic health indicators for the server process
- The current state of the JPS, such as the current network connections or recent performance statistics

For information about using the SRC CLI to monitor the JPS, see *Chapter 14, Monitoring the JPS with the SRC CLI*.

For information about using the C-Web interface to monitor the JPS, see *Chapter 15, Monitoring the JPS with the C-Web Interface*.

Chapter 14

Monitoring the JPS with the SRC CLI

This chapter describes how to use the SRC command-line interface (CLI) to monitor the Juniper Policy Server (JPS), a component of the SRC software that acts as a policy server in the PacketCable Multimedia Specification (PCMM) environment. You can use the CLI to monitor the JPS on a Solaris platform or on a C-series platform.

You can also use the C-Web interface to monitor the JPS. See *Chapter 15, Monitoring the JPS with the C-Web Interface*.

Topics include:

- Monitoring the JPS on page 147
- Viewing Server Process Information on page 147
- Viewing JPS State on page 148

For information about the JPS, see *Chapter 11, Using PCMM Policy Servers*.

Monitoring the JPS

You can monitor:

- The basic health indicators for the server process
- The current state of the JPS, such as the current network connections or recent performance statistics

To view information about the server process and the current state of the JPS:

user@host> **show jps statistics**

Viewing Server Process Information

To view information about the server process:

user@host> **show jps statistics process**

Viewing JPS State

You can monitor the current state of the JPS by:

- Viewing Performance Statistics for the JPS Interfaces on page 148
- Viewing Network Connections for the Application Manager on page 148
- Viewing Network Connections for the CMTS Device on page 148
- Viewing Performance Statistics for the CMTS Locator on page 149
- Viewing Message Handler Information on page 149

Viewing Performance Statistics for the JPS Interfaces

To view recent performance statistics for the application manager–to–policy server interface:

user@host> **show jps statistics am**

To view recent performance statistics for the policy server–to–CMTS interface:

user@host> **show jps statistics cmts**

To view recent performance statistics for the policy server–to–RKS interface:

user@host> **show jps statistics rks**

Viewing Network Connections for the Application Manager

To view information about the current JPS network connections for all the application managers:

user@host> **show jps statistics am connections**

To view information about the current JPS network connections for a specific application manager:

user@host> **show jps statistics am connections ip-address** *ip-address*

Enter all or part of the IP address to list connections for all matching addresses.

Viewing Network Connections for the CMTS Device

To view information about the current JPS connections for all the CMTS devices:

user@host> **show jps statistics cmts connections**

To view information about the current JPS connections for a specific CMTS device:

user@host> **show jps statistics cmts connections ip-address** *ip-address*

Enter all or part of the IP address to list connections for all matching addresses.

Viewing Performance Statistics for the CMTS Locator

To view information about the recent performance statistics for the CMTS locator:

user@host> **show jps statistics cmts-locator**

Viewing Message Handler Information

To view information about the JPS message handler and message flows:

user@host> **show jps statistics message-handler**
user@host> **show jps statistics message-handler message-flow**

To view information about specific JPS message flows:

user@host> **show jps statistics message-handler message-flow id** *id*

Enter all or part of the message flow identifier to list all matching message flows.

Chapter 15

Monitoring the JPS with the C-Web Interface

This chapter describes how to use the C-Web interface to monitor the Juniper Policy Server (JPS), a component of the SRC software that acts as a policy server in the PacketCable Multimedia Specification (PCMM) environment. You can use the C- interface to monitor the JPS on a Solaris platform or on a C-series platform.

This chapter contains the following topics:

- Viewing Information About JPS Server Process with the C-Web Interface on page 151
- Viewing JPS AM Statistics with the C-Web Interface on page 152
- Viewing JPS AM Connections with the C-Web Interface on page 153
- Viewing JPS CMTS Statistics with the C-Web Interface on page 153
- Viewing JPS CMTS Connections with the C-Web Interface on page 154
- Viewing JPS CMTS Locator Statistics with the C-Web Interface on page 155
- Viewing JPS Message Handler Statistics with the C-Web Interface on page 155
- Viewing JPS Message Flow Statistics with the C-Web Interface on page 156
- Viewing JPS RKS Statistics with the C-Web Interface on page 157

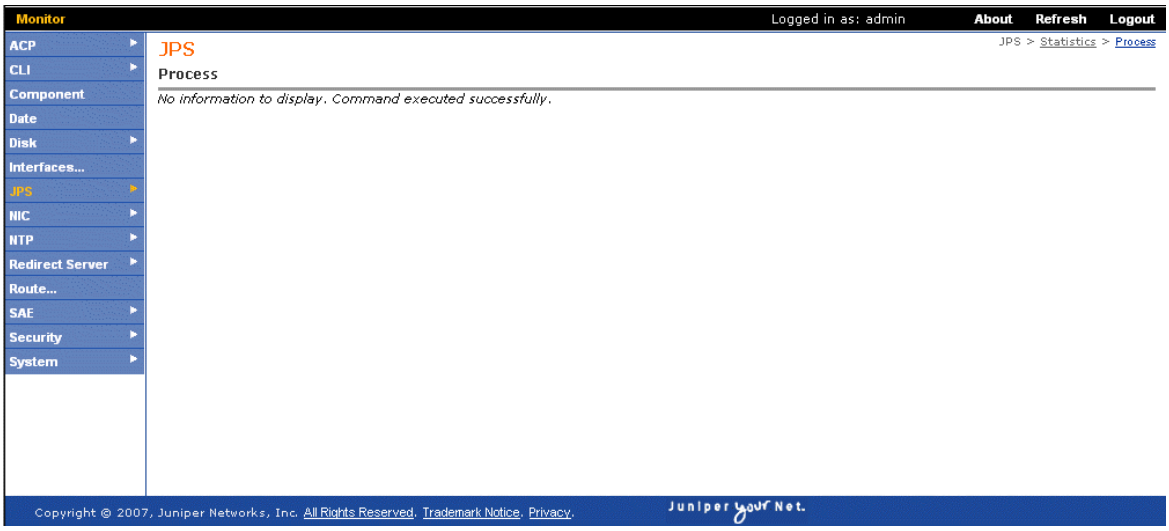
For information about the JPS, see *Chapter 11, Using PCMM Policy Servers*.

Viewing Information About JPS Server Process with the C-Web Interface

To view information about the JPS server process:

- Select **JPS** from the side pane, click **Statistics**, and then click **Process**.

The Process pane displays the JPS server process information.

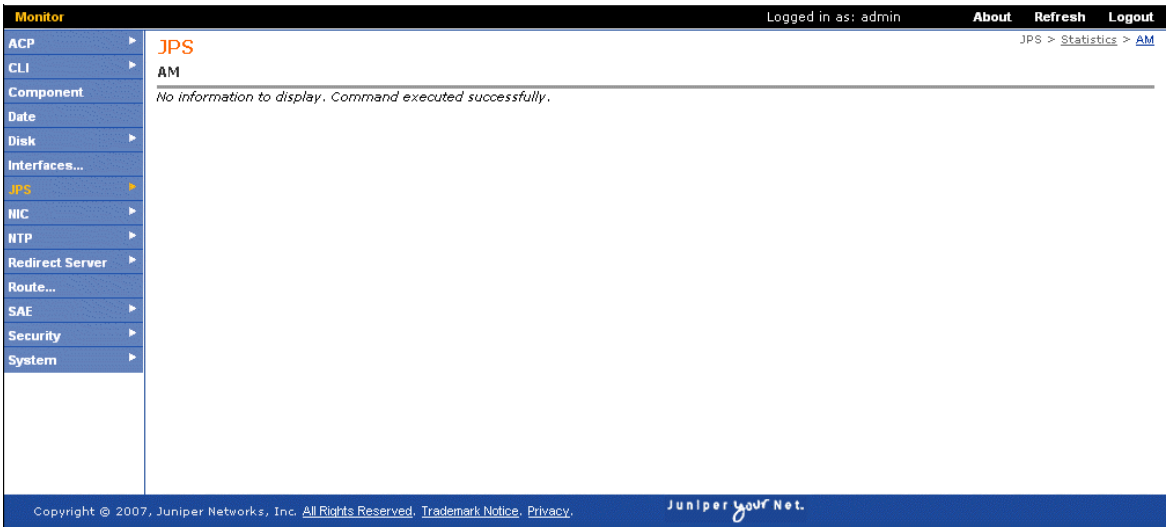


Viewing JPS AM Statistics with the C-Web Interface

To view information about recent performance statistics for the application manager–to–policy server interface:

- Select **JPS** from the side pane, click **Statistics**, and then click **AM**.

  The AM pane displays performance statistics for the application manager–to–policy server interface.



Viewing JPS AM Connections with the C-Web Interface

To view information about the current JPS network connections for the application manager:

1. Select **JPS** from the side pane, click **Statistics**, click **AM**, and then click **Connections**.

   The Connections pane appears.

2. In the IP Address box, enter the IP address, or leave the box blank to display all AM connections.

3. Click **OK**.

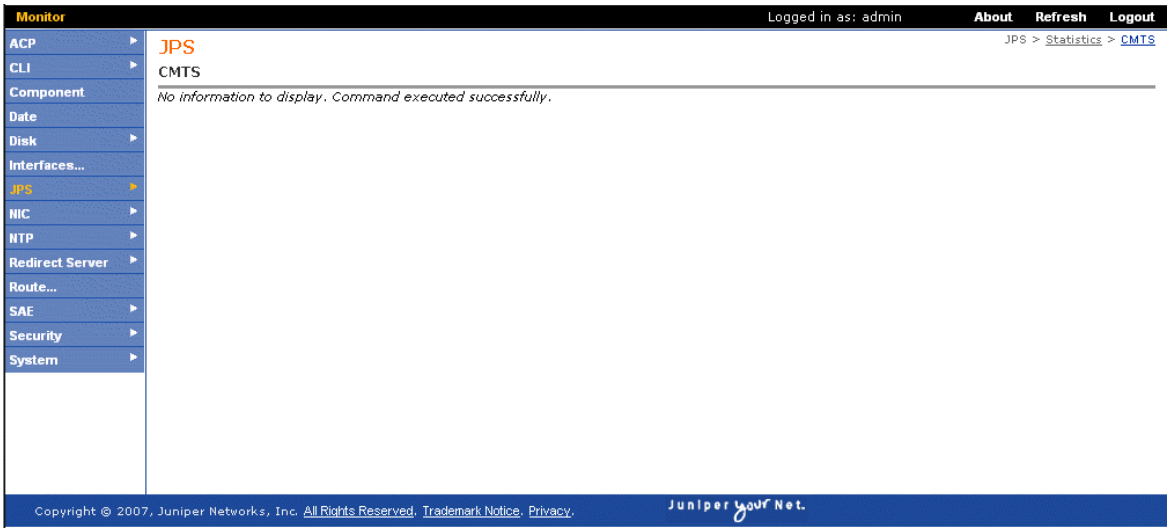
   The Connections pane displays the AM connection statistics.

Viewing JPS CMTS Statistics with the C-Web Interface

To view information about recent performance statistics for the policy server–to–CMTS interface:

- Select **JPS** from the side pane, click **Statistics**, and then click **CMTS**.

The CMTS pane displays statistics for the policy server–to–CMTS interface.



Viewing JPS CMTS Connections with the C-Web Interface

To view information about the current JPS network connections for the CMTS device:

1. Select **JPS** from the side pane, click **Statistics**, click **CMTS**, and then click **Connections**.

   The Connections pane appears.

2. In the IP Address box, enter the IP address, or leave the box blank to display all CMTS connections.

3. Click **OK**.

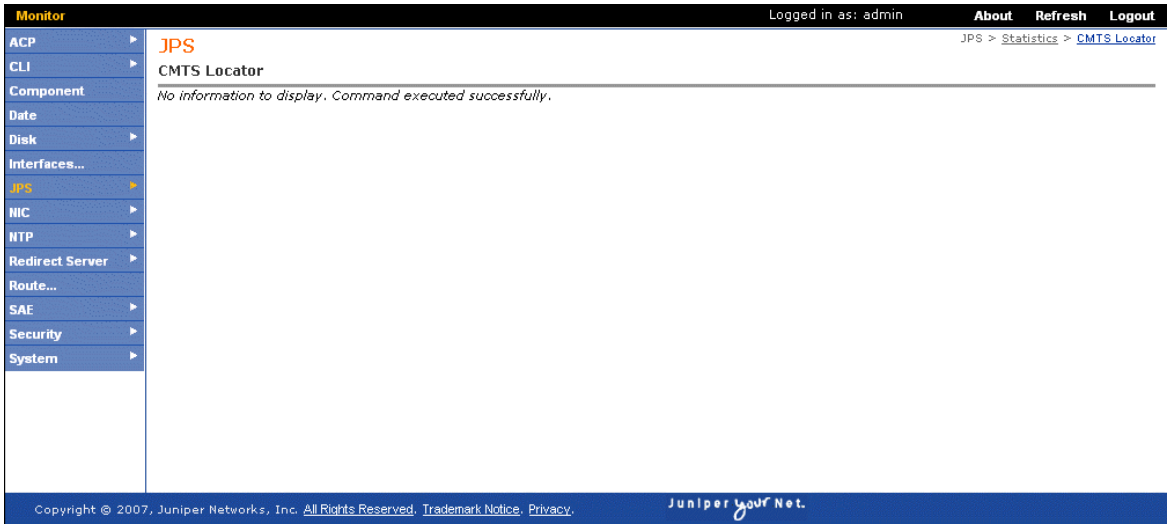
   The Connections pane displays the CMTS connection statistics.

Viewing JPS CMTS Locator Statistics with the C-Web Interface

To view information about the recent performance statistics for the CMTS locator:

- Select **JPS** from the side pane, click **Statistics**, and then click **CMTS Locator**.

  The CMTS Locator pane displays the CMTS locator statistics.

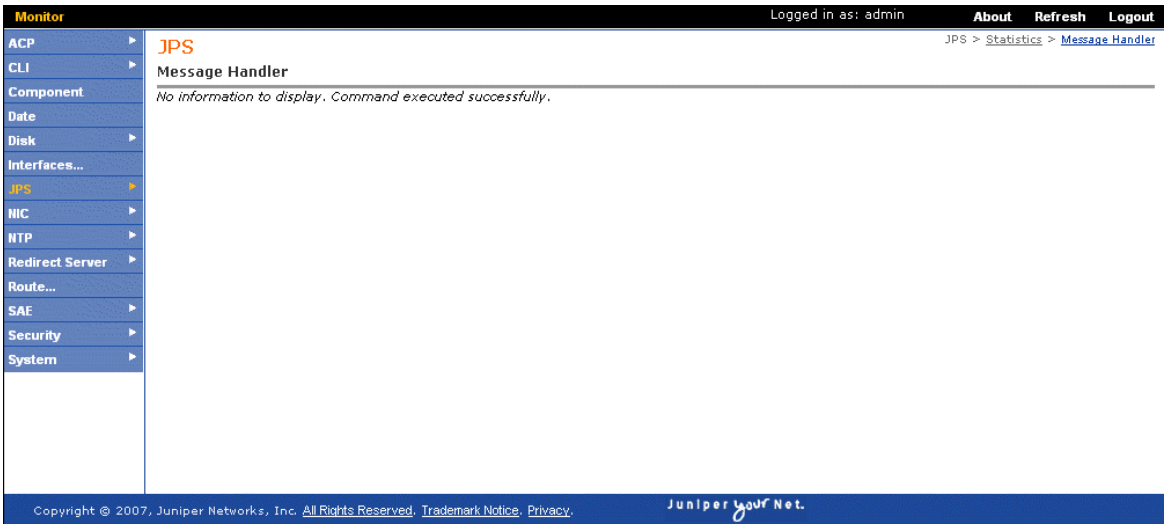


Viewing JPS Message Handler Statistics with the C-Web Interface

To view information about the JPS message handler:

- Select **JPS** from the side pane, click **Statistics**, and then click **Message Handler**.

The Message Handler pane displays the JPS message handler statistics.



Viewing JPS Message Flow Statistics with the C-Web Interface

To view information about JPS message flows:

1. Select **JPS** from the side pane, click **Statistics**, click **Message Handler**, and then click **Message Flows**.

   The Message Flows pane appears.

2. In the ID box, enter a message flow ID, or leave the box blank to display statistics for all message flows.

3. Click **OK**.

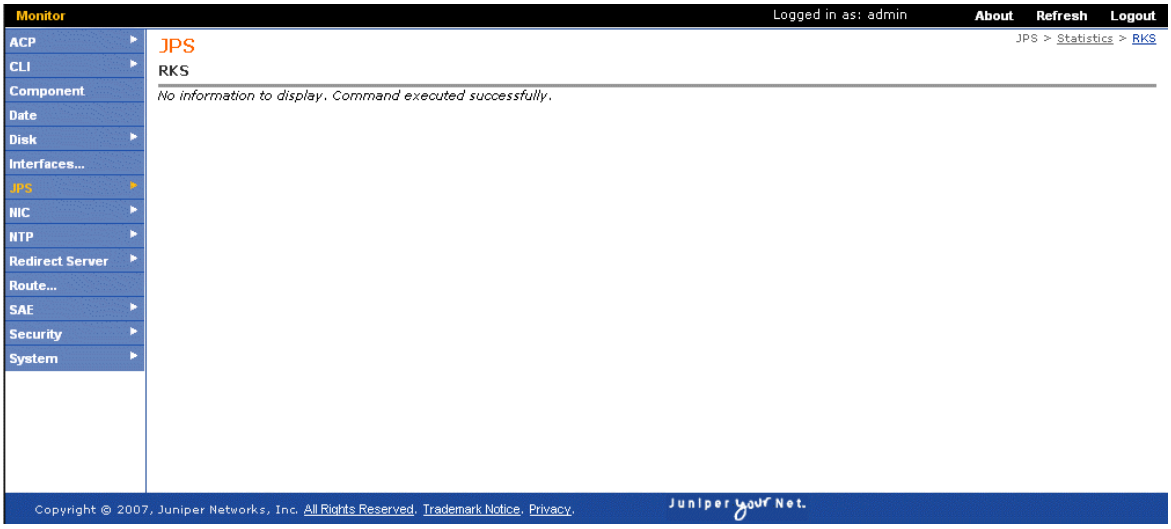
   The Message Flows pane displays the message flow statistics.

Viewing JPS RKS Statistics with the C-Web Interface

To view recent performance statistics for the policy server–to–record keeping server (RKS) interface:

- Select **JPS** from the side pane, click **Statistics**, and then click **RKS**.

  The RKS pane displays statistics for the policy server–to–RKS interface.



Chapter 16

Providing Packet Mirroring in the SRC Network

This chapter describes how the SRC network handles traffic mirroring on a JUNOSe router and how to configure policies, services, and subscribers that support RADIUS-based packet mirroring. Topics include:

- Overview of Packet Mirroring on page 159
- Configuring Packet Mirroring on page 160
- Specifying Maximum Number of Peers on page 164
- Example: Using the Sample Packet-Mirroring Application on page 164
- Defining RADIUS Attributes for Dynamic Authorization Requests with the API on page 166

Overview of Packet Mirroring

Packet mirroring allows you to mirror subscriber traffic by configuring a script service with the SRC software that applies policies on a JUNOSe router for RADIUS-based packet mirroring.

When the SAE activates a packet-mirroring service session, the session sends dynamic RADIUS requests, such as change-of-authorization (CoA) messages, to a RADIUS device such as a JUNOSe router.

In RADIUS-based packet mirroring on a JUNOSe router, a RADIUS administrator uses RADIUS attributes to configure packet mirroring of a particular subscriber's traffic. The router creates dynamic secure policies for the mirroring operation. The original traffic is sent to its intended destination, and the mirrored traffic is sent to an analyzer device (the mediation device). The mirroring operations are transparent to the subscriber whose traffic is being mirrored. This dynamic method uses RADIUS attributes and RADIUS vendor-specific attributes (VSAs) to identify a subscriber whose traffic is to be mirrored and to trigger the mirroring session. RADIUS-based mirroring uses dynamically created secure policies based on certain RADIUS VSAs. You attach the secure policies to the interface used by the mirrored subscriber. The packet-mirroring VSAs that the RADIUS server sends to the E-series router are MD5 salt-encrypted.

You must deploy RADIUS-based packet mirroring on JUNOSe routers to monitor the subscriber traffic.

Configuring Packet Mirroring

To support packet mirroring in an SRC network, configure a script service that can be activated to set up RADIUS-based packet-mirroring policies on a JUNOSe router. The script service defines the parameters needed to mirror subscriber traffic, such as the address of the subscriber or the analyzer device. This script service is activated for the subscriber whose traffic should be mirrored. For detailed information about configuring script services, see *SDX Services and Policies Guide, Chapter 2, Managing Services on a Solaris Platform*.

You must have preconfigured RADIUS-based packet mirroring on JUNOSe routers. The JUNOSe software provides RADIUS-based packet mirroring, which allows the router to create dynamic secure policies for the mirroring operation. The RADIUS administrator can configure and manage interface mirroring services that are activated by means of CoA. For information about configuring RADIUS-based packet mirroring on the JUNOSe router, see the *JUNOSe Policy Management Configuration Guide*.

For information about dynamic RADIUS requests, see RFC 3576—Dynamic Authorization Extensions to Remote Authentication Dial In User Service (RADIUS) (July 2003).

To set up the SRC software for packet mirroring, perform the following tasks:

- Creating the Script Service for Packet Mirroring on page 160
- Configuring the Script Service for Packet Mirroring on page 162
- Configuring Subscriptions to the Packet-Mirroring Service on page 164
- (Optional) Specifying Maximum Number of Peers on page 164

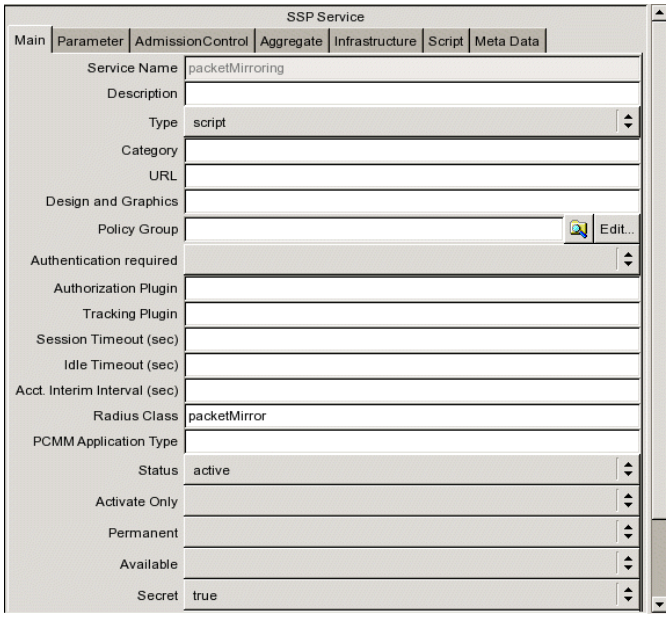
The SRC software includes a sample script service that you can configure to send dynamic RADIUS requests to the JUNOSe router. You can use the sample service definition and customize it for your environment by modifying the service substitutions. For information about the sample packet mirroring application, see *Example: Using the Sample Packet-Mirroring Application* on page 164.

Creating the Script Service for Packet Mirroring

To create the script service:

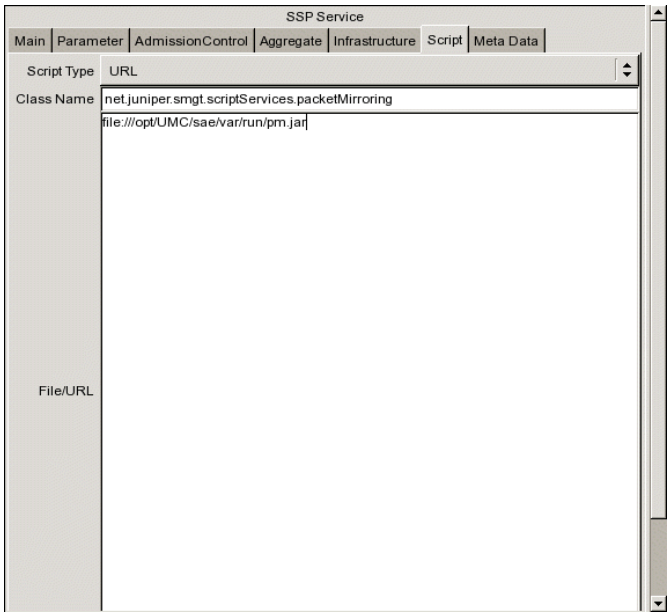
1. In the SDX Admin navigation pane, right-click the Services folder, highlight **New**, and then click **SSP Service**.
2. In the New SSP Service dialog box, enter a service name or select a name from the drop-down list.
3. In the Main tab pane, select **script** in the Type field.

4. If you want to hide the service from users and unauthorized administrators, select **true** from the menu in the Secret field.



5. Click the Script tab.

   The Script pane appears.



6. Edit the values in the Script fields for the sample packet-mirroring script service.

   - In the Script Type field, select **URL**.
   - In the Class Name field, enter net.juniper.smgt.sae.packetMirroring.LiService.
   - In the File/URL field, enter *file:///opt/UMC/sae/var/run/pm.jar*.

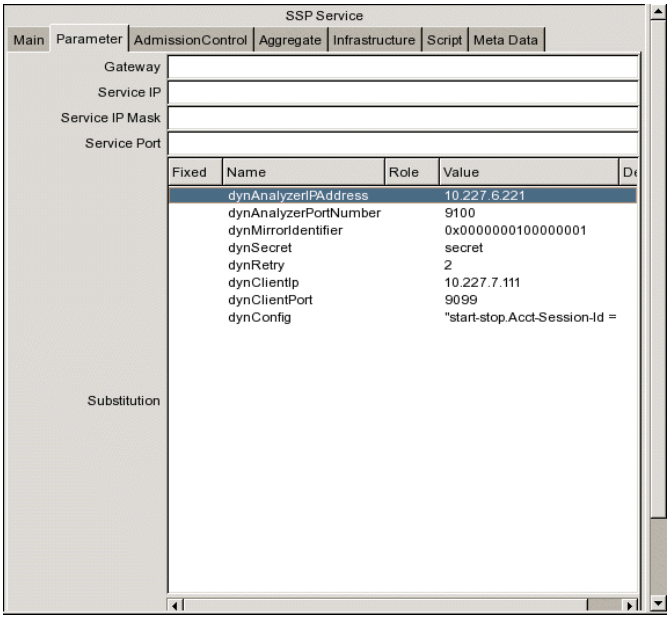
7. Click **Save**.

After you create the script service, you need to configure parameters for the script service. For more information about configuring script services and parameters, see *SDX Services and Policies Guide, Chapter 2, Managing Services on a Solaris Platform*.

Configuring the Script Service for Packet Mirroring

To configure the script service, you provide parameter substitutions with the values that are in the service definitions. To do so:

1. In SDX Admin, select the Parameter tab in the script service configuration. The parameter pane appears.



2. Configure the parameters.

   Table 9 lists the parameters specified by the sample packet-mirroring script service. In most cases, you can use the sample script service without modification.

Table 9: Parameter Substitutions for Packet-Mirroring Services

Parameter Name	Description
dynAnalyzerIPAddress	RADIUS VSA that is the IP address of the analyzer device. This attribute is required.
dynAnalyzerPortNumber	RADIUS VSA that is the UDP port number of the monitoring application in the analyzer device. If specified, dynMirrorIdentifier must also be specified.
dynMirrorIdentifier	RADIUS VSA in the form of a hexadecimal string. If specified, dynAnalyzerPortNumber must also be specified.
dynClientIp	IP address of the dynamic RADIUS client.
dynClientPort	UDP port number of the dynamic RADIUS client.
dynSecret	Shared secret.
dynRetry	Number of retries for sending dynamic RADIUS packet when no RADIUS response is received. The retry interval is 3 seconds.
dynConfig	Content of dynamic RADIUS request packets in the format <action>. <radiusAttributeName> = <pluginEventAttribute>\n ■ action—Action that is executed on packet content (attribute) ■ start ■ stop ■ start-stop ■ radiusAttributeName—Valid RADIUS attribute specified as follows: ■ Standard RADIUS attribute name or number. ■ JUNOSe VSA in one of the following formats: vendor-specific.4874.<vsa#>[.salt] 26.4874.<vsa#>[.salt] where .salt indicates that the attribute is MD5 salt-encrypted in the RADIUS packet. ■ pluginEventAttribute—Valid Python expression ■ \n—New-line character included between the lines of a configuration containing multiple lines; the entire configuration must be enclosed in quotation marks For example: start-stop.Acct-Session-Id = ifSessionId "start-stop.Acct-Session-Id = ifSessionId\nstart.vendor-specific.4874.58.salt = 1\nstart.vendor-specific.JUNIPER.Unisphere-Med-Dev-Handle.salt = custom['dynMirrorIdentifier']\nstart.vendor-specific.JUNIPER.Unisphere-Med-Ip-Address.salt = intIp(custom['dynAnalyzerIPAddress'])\nstart.vendor-specific.JUNIPER.Unisphere-Med-Port-Number.salt = int(custom['dynAnalyzerPortNumber'])\nstop.vendor-specific.4874.58.salt = 0"

You can also configure dynamic RADIUS requests with the sendDynamicRadius method of the ServiceSessionInfo interface (see *Defining RADIUS Attributes for Dynamic Authorization Requests with the API* on page 166).

For detailed information about configuring services, see *SDX Services and Policies Guide, Chapter 2, Managing Services on a Solaris Platform*.

Configuring Subscriptions to the Packet-Mirroring Service

You need to configure subscriptions to the packet-mirroring service. You can set up the subscriptions to activate immediately on login.

For more information, see *SDX Subscribers and Subscriptions Guide, Chapter 13, Configuring Subscribers and Subscriptions with SDX Admin*.

Specifying Maximum Number of Peers

The dynamic RADIUS server can maintain a certain number of peers.

To specify the maximum number of peers with SDX Configuration Editor:

1. In the navigation pane, select a directory configuration object for the SAE that you want to configure.
2. Select the Miscellaneous tab, and expand the Dynamic Radius Server section.

3. Edit or accept the default value in the Maximum Cached Peer field.

Maximum Cached Peer

- Maximum number of peers maintained by the dynamic RADIUS server.
- Value—Integer
- Default—100
- Property name—dynRadius.peers.cacheSize

Example: Using the Sample Packet-Mirroring Application

To use the sample packet-mirroring application provided:

1. Import the sample service definition using an LDAP browser.

   The */SDK/scriptServices/packetMirroring/ldif/service.ldif* file (in the SRC software distribution) is the sample service definition.

2. Copy the */lib/pm.jar* file used by the script service to the */var/run* directory in the SAE installation directory (*/opt/UMC/sae* by default).

3. Modify the service substitutions for your environment.

   You can make these substitutions by defining the parameter substitutions in the packetMirroring service (*serviceName = packetMirroring, o = Services, o = umc*) with SDX Admin or by passing the values through the SAE core API.

For information about parameter substitutions, see *Configuring the Script Service for Packet Mirroring* on page 162. For information about passing the values through the SAE core API, see *Defining RADIUS Attributes for Dynamic Authorization Requests with the API* on page 166.

4. Configure a subscription to the packetMirroring service that is activated on login.

   For more information about subscriptions, see *SDX Subscribers and Subscriptions Guide, Chapter 13, Configuring Subscribers and Subscriptions with SDX Admin*.

If you are modifying the sample application, add the *sae.jar* and *logger.jar* files to the classpath when you compile your application. These two files can be found in the *lib* directory of the SAE installation directory.

Example: Packet Mirroring for PPP Subscribers

When a PPP subscriber is subscribed to the packet-mirroring service, the service should be configured as an activate-on-login service at user connection time. After the subscriber has logged in through the SAE remote API, the packet-mirroring service can be subscribed to the PPP subscriber and activated. When the service is activated, a CoA request is sent to the JUNOSe router that includes the PPP subscriber's accounting session ID to start packet mirroring for this subscriber.

Example: Packet Mirroring for DHCP Subscribers

When a DHCP subscriber is subscribed to the packet-mirroring service, the service should be configured as an activate-on-login service at user connection time. After the subscriber has logged in through the SAE remote API, the packet-mirroring service can be subscribed to the DHCP subscriber and activated. When the service is activated, a CoA request is sent to the JUNOSe router that includes the DHCP subscriber's IP address and virtual router name for the JUNOSe router to start packet mirroring for this subscriber.

Configuring DHCP Subscriber Sessions

You can use DHCP option 82 to identify the subscriber session. For example, if you set DHCP option 82 as the user login name, an external application can use this setting to search for the subscriber session. The following subscriber classification script illustrates this example:

```
[retailername=default,o=Users,o=UMC?loginName=<-dhcp[82].suboptions[1].string->?
sub?(interfaceName=<-dhcp[82].suboptions[1].string->)]
loginType = “ADDR”

[<-retailerDN->??sub?(uniqueID=<-userName->)]
retailerDN != ““
& userName != ““

[<-unauthenticatedUserDn->]
loginType == "ADDR"
loginType == "AUTHADDR"
```

Disabling RADIUS Authentication for DHCP Subscribers

Packet mirroring for DHCP subscribers does not involve RADIUS authentication, so you might have to configure authentication to grant all IP subscriber management interfaces access without authentication. For example, configure the JUNOSe router with the following authentication:

```
aaa authentication ip default none
```

You can still configure other subscribers to use RADIUS authentication. For example, configure the JUNOSe router with the following authentication for PPP subscribers:

```
aaa authentication ppp default radius
```

Defining RADIUS Attributes for Dynamic Authorization Requests with the API

The SRC software provides two ways to define RADIUS attributes for dynamic RADIUS authorization requests:

- Service definition (see *Configuring the Script Service for Packet Mirroring* on page 162)
- SAE core API

NOTE: Parameters set in the API override parameters set by the service definition.

To send dynamic RADIUS authorization requests with the SAE core API, the script service uses the sendDynamicRadius and getRouterDynRadiusAddr methods in the ServiceSessionInfo interface to provide the content of the RADIUS packet for the dynamic authorization request to the JUNOSe router that is attached to the service session.

For information about the ServiceSessionInfo interface, see the script service documentation in the SRC software distribution in the folder *SDK/doc/sae* or in the SAE core API documentation on the Juniper Networks Web site at

http://www.juniper.net/techpubs/software/management/sdx/api-index.html

For a sample implementation, see the following file in the SRC software distribution:

SDK/scriptServices/packetMirroring/java/net/juniper/smgt/scriptServices/packetMirroring/LiService.java

Chapter 17

Configuring IPTV Services in an SRC Network

This chapter describes how the SRC network handles IP television (IPTV) services, and how to configure policies, services, and subscribers that support IPTV applications. It contains the following sections:

- Overview of IPTV Service Applications on page 167
- Installing the Sample IPTV Application on page 168
- Configuring the Sample IPTV Application on page 168
- Running the Sample IPTV Application on page 176

Overview of IPTV Service Applications

IPTV and multimedia service sessions are typically established in multiple phases that require changes to installed policies and authorized bandwidth while the service session remains active. To support IPTV sessions, the SAE allows changes to active service sessions. These changes include:

- Controlled bandwidth. If bandwidth demand increases, the authorization plug-in must authorize the change.
- Policy parameters. Only parameter substitution values can be changed. Policy parameters can include classifiers, such as destination address, and actions, such as rate-limit profiles.
- Session and idle timeouts. All attributes that can be set for initial service activation can be set for service session modifications.

You can integrate IP servers into an SRC-managed network so that the SRC software can receive network resource requests from IPTV applications to perform admission control and install policies. The SRC software can also set up and manage the initial label-switched paths (LSPs) between the video servers and the edge routers.

When the SAE activates a service session, it authorizes the session with authorization plug-ins; it may use the SRC Admission Control Plug-in (SRC-ACP) application to perform call admission control (CAC) and allocate bandwidth; and it installs the policy required for the service on a router interface.

To support IPTV services, we provide a sample IPTV application that uses the extended capabilities of SRC-ACP and the SAE. You can use SRC-ACP to initialize and execute applications that are attached to congestion points. You can use the SAE router driver to set up and manage LSPs.

Installing the Sample IPTV Application

You must manually install the UMCiptv package on the server host to deploy the sample IPTV application.

```
pkgadd -d /cdrom/cdrom0/solaris UMCiptv
```

NOTE: The sample IPTV application is provided on the SRC application library CD.

For information about installing the sample IPTV application, see *SRC Application Library Guide, Chapter 1, Installing the SRC Applications*.

Configuring the Sample IPTV Application

The SRC application library provides a sample IPTV application with sample data that you must modify for your environment. The sample data provides the minimum requirements for demonstrating an IPTV application in the SRC-managed network. Figure 19 on page 170 illustrates a sample network configuration that contains Juniper Networks routers.

The sample network configuration has the following setup:

- The IPTV server connects to the M-series routing platform.
- The SAE manages the M-series routing platform.
- The SAE manages the E-series router and all its subscribers.
- The NIC is configured for the SAE.
- SRC-ACP is configured in backbone mode for the SAE managing the M-series routing platform.
- The M-series routing platform and the E-series router have MPLS configured for creating the LSP tunnel between them.

This sample configuration allows the sample IPTV application to:

- Start the IPTV service.

  When the subscriber requests a video stream from the IPTV server, the IPTV service is activated by the SAE, and SRC-ACP activates a script service to create the LSP tunnel between the two routers over which the video stream is sent. SRC-ACP also updates the bandwidth usage for the LSP tunnel so the IPTV server can send the video stream to the subscriber.

- Stop the IPTV service.

  When the subscriber is finished viewing the video stream from the IPTV server, the IPTV service is deactivated by the SAE, and SRC-ACP activates a script service to remove the tunnel.

- Manage tunnel bandwidth.

  When tunnel bandwidth usage reaches the incremental threshold (80%) of the initial bandwidth value, SRC-ACP modifies the script service on the SAE to increase the bandwidth to the incremental bandwidth value. When the tunnel bandwidth usage reaches the decremental threshold (20%), SRC-ACP modifies the script service to decrease the bandwidth to the initial bandwidth value.

- Monitor tunnel state.

  When the LSP tunnel is down, SRC-ACP can report the affected IPTV service sessions to the IPTV server.

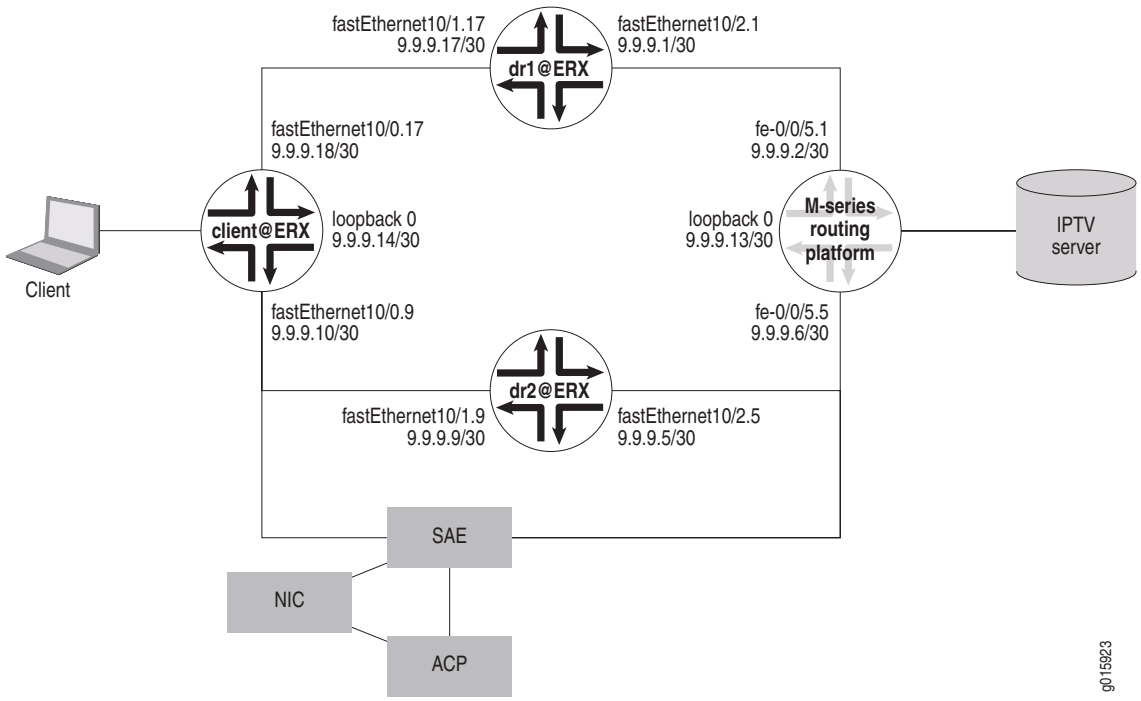
This section describes the configuration tasks for configuring the sample IPTV application.

- Setting Up the IPTV Network on page 170
- Configuring the SAE for the IPTV Application on page 171
- Configuring SRC-ACP for the IPTV Application on page 173
- Configuring the NIC for the IPTV Application on page 176

Setting Up the IPTV Network

Set up the network shown in Figure 19. The sample routing configuration is based on this network configuration.

Figure 19: Sample IPTV Network Configuration



To set up routing configuration from the IPTV client to the IPTV server, modify the following sample router scripts with interface names that match your network device names.

- */iptv/conf/network/client.scr*—Sets up the LSP tunnel endpoint (JUNOSe router)
- */iptv/conf/network/dr1.scr* — Sets up the path for the LSP tunnel (JUNOSe router)
- */iptv/conf/network/dr2.scr* — Sets up the path for the LSP tunnel (JUNOSe router)
- */iptv/conf/network/lsptest.scr* — Sets up the LSP tunnel starting point (JUNOS routing platform)

These scripts do not create the LSP tunnel. The tunnel service creates the LSP tunnel when it is activated. See *Configuring IPTV Subscribers and Services* on page 171.

Configuring the SAE for the IPTV Application

You must configure the SAE to:

1. Manage the routers.
2. Configure subscribers and services.
3. Configure SRC-ACP as an external plug-in for the SAE.
4. Configure event publishers.
5. Configure the NIC as an external plug-in.

Managing the Routers in an IPTV Network

You must configure the SAE to manage the JUNOSe router connected to the IPTV subscriber (client@ERX) and the JUNOS routing platform connected to the IPTV server. The JUNOSe router must be configured to manage the LSP tunnel so that the SAE can report information about the LSP tunnel to SRC-ACP.

To configure the SAE to manage the JUNOSe router or the JUNOS routing platform, see *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 6, Using JUNOSe Routers in the SRC Network with a Solaris Platform* or *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 8, Using JUNOS Routing Platforms in the SRC Network with a Solaris Platform*.

Configuring IPTV Subscribers and Services

Define the subscribers by importing the */iptv/conf/ldap/subscribers.ldif* file from the installation directory (*/opt/UMC* by default) using an LDAP browser. You can then modify the following two subscribers with SDX Admin.

- *uniqueID = iptvSubscriber, ou = local, retailermame = IPTV, o = Users, o = umc*

  The iptvSubscriber subscriber is the IPTV client.

- *uniqueID = iptvServer, ou = local, retailermame = IPTV, o = Users, o = umc*

  The iptvServer subscriber is the IPTV server.

For detailed information about configuring subscribers, see *SDX Subscribers and Subscriptions Guide, Chapter 13, Configuring Subscribers and Subscriptions with SDX Admin*.

Define the IPTV services by importing the */iptv/conf/ldap/services.ldif* file from the installation directory (*/opt/UMC* by default) using an LDAP browser. You can then modify the services with SDX Admin.

- *serviceName = iptv, o = Services, o = umc*

  The service named iptv is a policy-driven service with the default policy that forwards any traffic. The iptv service allows the IPTV subscriber to allocate bandwidth. The iptv service should be linked to the active congestion point so that SRC-ACP can perform CAC. The bandwidth and congestion point for the service are specified in the Admission Control tab of the SSP Service pane.

- *serviceName = tunnel, o = Services, o = umc*

  The tunnel service is a script service that creates an LSP tunnel. The script service must specify an idle timeout value and a substitution for the IPTV_LSPTunnel_ingress attribute (which is the IP address of the LSP tunnel ingress). The script service uses the getCommandChannel method in the ServiceSessionInfo interface to provide the command channel on which to send commands to the network device (such as a Juniper Networks router) that is attached to the service session. For the script service to work, you must copy the */iptv/lib/iptv.jar* file to the */sae/var/run* directory.

  For information about the ServiceSessionInfo interface, see the script service documentation in the SRC software distribution in the folder *SDK/doc/sae* or in the SAE core API documentation on the Juniper Networks Web site at

  http://www.juniper.net/techpubs/software/management/sdx/api-index.html

For detailed information about configuring services, see *SDX Services and Policies Guide, Chapter 2, Managing Services on a Solaris Platform*. For detailed information about configuring policies, see *SDX Services and Policies Guide, Chapter 12, Configuring and Managing Policies with Policy Editor*.

NOTE: The SAE and JUNOSe routers communicate using the Common Open Policy Service (COPS) protocol. The sample data supports COPS XDR mode.

To use the sample data with COPS-PR mode, you must perform one of the following tasks:

- Define another IPTV subscriber that can start the service named iptv.
- Define another service (for example, iptv2) so that the same client can start two services.

Configuring SRC-ACP as an External Plug-In for the IPTV Application

To configure an external plug-in for the SAE, see *SDX Subscribers and Subscriptions Guide, Chapter 5, Configuring Subscriber-Related Properties on the SAE on a Solaris Platform*.

Configure the object reference with the following value:

- Plugin.acp.objectref = corbaname:: < host > : < port > /NameService#plugin.acp
  - < host > —Name or IP address of the COS name server
  - < port > —TCP port

Use the following values for the plug-in attributes:

- PA_ROUTER_NAME, PA_INTERFACE_NAME, PA_INTERFACE_ALIAS, PA_PORT_ID, PA_SERVICE_NAME, PA_EVENT_TIME, PA_SESSION_ID, PA_NAS_IP, PA_DOWNSTREAM_BANDWIDTH, PA_UPSTREAM_BANDWIDTH, PA_SERVICE_SESSION_NAME, PA_EVENT_TIME_MILLISECOND, PA_INTERFACE_SPEED, PA_SUBSTITUTION

Configuring Event Publishers for the IPTV Application

Configure the SAE to publish the following types of events to SRC-ACP:

- Global service tracking
- Global interface tracking

Identify the instance of SRC-ACP by the name of the host on which you configured it. For example:

- Service.tracking.plugins = acp
- Interface.tracking.plugins = acp

Configure the SAE to publish the global user tracking events to the NIC; for example: User.tracking.plugins = nic.

For information about configuring event publishers, see *SDX Subscribers and Subscriptions Guide, Chapter 11, Configuring Authorization and Accounting Plug-Ins with SDX Configuration Editor*.

Configuring the NIC as an External Plug-In for the IPTV Application

To configure an external plug-in for the SAE, see *SDX Subscribers and Subscriptions Guide, Chapter 10, Overview of Configuring Plug-Ins for Solaris Platforms*.

Configure the object reference with the following value:

Plugin.nic.objectref = corbaname:: < host > : < port > /NameService#nicsae/saePort

- < host > —Name or IP address of the COS name server
- < port > —TCP port

Use the following values for the plug-in attributes:

- PA_ROUTER_NAME, PA_SESSION_ID, PA_USER_TYPE, PA_USER_DN, PA_USER_IP_ADDR

Configuring SRC-ACP for the IPTV Application

To configure SRC-ACP for the sample IPTV application, you must configure SRC-ACP as you would normally for backbone mode and enable state synchronization. For more information about configuring SRC-ACP, see *SRC Application Library Guide, Chapter 20, Providing Admission Control with SRC-ACP on a Solaris Platform*.

In addition to the normal SRC-ACP configuration, you must perform these tasks:

1. Defining SRC-ACP Properties for the IPTV Application on page 174
2. Defining the Sample Congestion Points for the IPTV Application on page 175

Defining SRC-ACP Properties for the IPTV Application

To configure SRC-ACP for the sample IPTV application, you must define certain SRC-ACP properties. To do so with SDX Admin:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry *I = config, I = ACP, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*.
3. Click the Main tab in the ACP Configuration pane.
4. In the Property field, include these entries so that the NIC proxy in ACP can look up the SAE using the DN.

   ```
   /NicProxy.IPTV/nic.server = /realms/dn/B1
   /NicProxy.IPTV/nic.keytype = Dn
   /NicProxy.IPTV/nic.valuetype = SaeId
   /NicProxy.IPTV/nic.expectmultiple = false
   ```

   For information about NIC proxies, see *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 13, Configuring Applications to Communicate with an SAE*.

5. In the Property field, include these entries to specify how SRC-ACP manages the congestion points for the IPTV application and to specify the ACP configuration namespace.

   ```
   Application.iptv.tunnelaction.retryInterval = 5
   Application.iptv.tunnelaction.timeout = 10
   Application.iptv.tunnelaction.maxRetry = 2
   Application.iptv.tunnelaction.waitTime = 60
   Application.iptv.tunnelaction.nicNamespace = /<-acpnamespace->/NicProxy.IPTV
   ```

Application.iptv.tunnelaction.retryInterval

- Time interval at which SRC-ACP tries to create, delete, or modify the LSP tunnel.
- Value—Number of seconds in the range 0–2147483647
- Default—Application.iptv.tunnelaction.retryInterval = 5

Application.iptv.tunnelaction.timeout

- Maximum time SRC-ACP waits to create, delete, or modify the LSP tunnel.
- Value—Number of seconds in the range 0–2147483647
- Default—Application.iptv.tunnelaction.timeout = 10

Application.iptv.tunnelaction.maxRetry

- Maximum number of retry times for creating, deleting, or modifying the LSP tunnel.
- Value—Number
- Default—Application.iptv.tunnelaction.maxRetry = 2

Application.iptv.tunnelaction.waitTime

- Time to wait before re-creating the LSP tunnel after an interface tracking event reports that the LSP tunnel is down.
- Value—Number of seconds in the range 0–2147483647
- Default—Application.iptv.tunnelaction.waitTime = 60

Application.iptv.tunnelaction.nicNamespace

- Name of the object that contains the SRC-ACP configuration data for the application.
- Value—/ < ACP namespace > /NicProxy.IPTV
- Guidelines—Replace < -acpnamespace- > with the SRC-ACP configuration namespace. This object appears in *l = ACP, ou = staticConfiguration, o = Management, o = umc*.
- Default—Application.iptv.tunnelaction.nicNamespace = / < -acpnamespace- > /NicProxy.IPTV
- Example—Application.iptv.tunnelaction.nicNamespace = /jacp/NicProxy.IPTV

Defining the Sample Congestion Points for the IPTV Application

Define the sample network congestion points and the sample congestion point for the service named iptv by importing the */iptv/conf/ldap/congestionPoint.ldif* file from the installation directory (*/opt/UMC* by default) using an LDAP browser. You can then modify the following congestion points with SDX Admin.

- The congestion points for the iptv service are added under *o = congestionPoint*, *o = umc*.
- The network congestion points are added under *o = admissionControl*, *o = umc*.

  The *interfaceName = iptv*, *orderedCimKeys = client@ERX*, *o = admissionControl*, *o = umc* object contains a link to the application. For the link to work, you must copy the */iptv/lib/iptv.jar* file to the */acp/var/run* directory.

  The sample data for the active congestion point includes the following parameters for managing the LSP tunnel's bandwidth and endpoint information based on the sample application:

  - CP_LSP_Egress_IP—Egress IP address for LSP tunnel
  - CP_LSP_Initial_Bandwidth—Initial bandwidth for LSP tunnel (in bps)
  - CP_LSP_Max_Bandwidth—Maximum bandwidth for LSP tunnel (in bps)
  - CP_LSP_Incremental_Bandwidth—Incremental bandwidth for LSP tunnel after the threshold is triggered (in bps)
  - CP_LSP_Name—LSP tunnel name

- CP_LSP_DecrementBandwidth_Threshold—LSP decrement bandwidth threshold in percentage
- CP_LSP_IncrementBandwidth_Threshold—LSP increment bandwidth threshold in percentage

The SRC software can trigger changes to the bandwidth of the LSP tunnel based on a given threshold. For example, bandwidth can be increased by 50 Mbps if 90% of the initial bandwidth is utilized by setting these parameter values:

```
CP_LSP_Incremental_Bandwidth = 50000000
CP_LSP_IncrementBandwidth_Threshold = 0.9
```

Your parameters can differ if you write your own application.

For information about configuring congestion points, see *SRC Application Library Guide, Chapter 20, Providing Admission Control with SRC-ACP on a Solaris Platform*.

Configuring the NIC for the IPTV Application

To configure the NIC for the sample IPTV application:

- Configure a NIC proxy. See *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 13, Configuring Applications to Communicate with an SAE*.
- Add the NIC configuration entry *l = OnePopAllRealms*, *l = NIC*, *ou = staticConfiguration*, *ou = Configuration*, *o = Management*, *o = umc* by importing the */iptv/conf/ldap/nic.ldif* file from the installation directory (*/opt/UMC* by default) using an LDAP browser.

  This entry contains a special DN to SAE ID mapping. The NIC host configuration namespace must point to this entry.

Running the Sample IPTV Application

To run the sample IPTV application, you must:

1. Install the omniORB and SMCpython packages from the SRC software distribution. See the *SDX Getting Started Guide* for complete information about installing the SRC core software.
2. Start the SAE, the NIC, and SRC-ACP, and establish the router connections.
3. Log in to the IPTV server as a static interface user on the JUNOS routing platform, and log in as the IPTV subscriber on the JUNOSe router.
4. In the */bin* directory, compile the SAE IDL by running compileIDL.
5. Include the omniORB library (*/opt/UMC/omni/lib* by default) in the LD_LIBRARY_PATH environment. For example:

   ```
   export LD_LIBRARY_PATH=/opt/UMC/omni/lib:$LD_LIBRARY_PATH
   export PATH=/opt/UMC/python/bin:$PATH
   ```

6. In the */bin* directory, run the *startIPTVService1.py* Python script.
7. In the */bin* directory, run the *startIPTVService2.py* Python script.
8. In the */bin* directory, run the *stopIPTVService2.py* Python script.
9. In the */bin* directory, run the *stopIPTVService1.py* Python script.

Chapter 18

Providing Services in IMS Networks

This chapter describes the SRC application's support for IP multimedia subsystem (IMS). Topics include:

- Overview of an IMS Environment on page 179
- IMS and ETSI References on page 180
- IMS Layers on page 181
- ETSI-TISPAN Architecture on page 183
- SRC Software in the ETSI-TISPAN Architecture on page 185
- SRC Software in the IMS Environment on page 186
- Installing and Configuring the IMS Software on page 187
- Testing and Demonstrating the A-RACF Rq Interface on page 192
- Configuring Policies for IMS on page 193

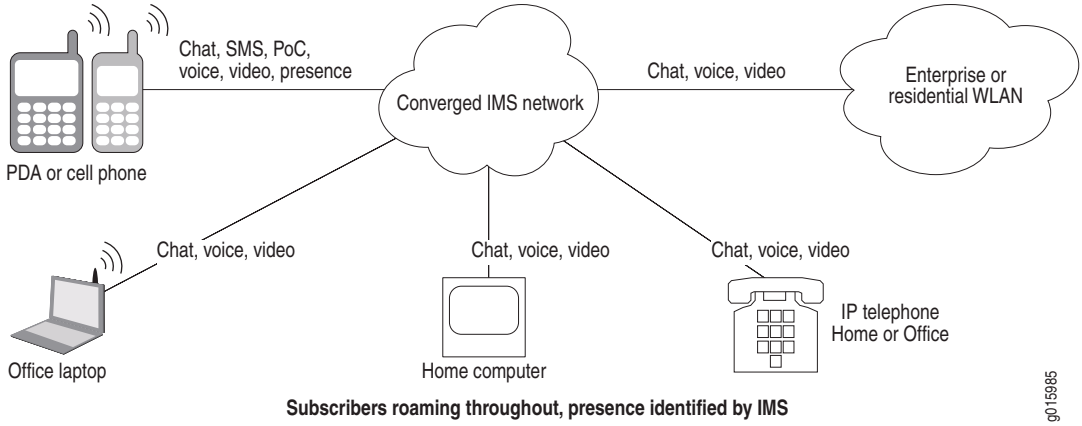
Overview of an IMS Environment

IMS is a flexible network architecture that allows providers to introduce rich multimedia services across both next-generation packet-switched and traditional circuit-switched networks. It uses open interfaces and functional components that can be assembled flexibly to support real-time interactive services and applications.

Third Generation Partnership Project (3GPP) developed IMS to provide a standards-based architecture for mobile carriers to migrate to their next-generation networks that will support applications that combine voice, video, and data functionality. The European Telecommunications Standards Institute (ETSI) created Telecommunications and Internet Converged Services and Protocols for Advanced Networks (TISPAN) to extend IMS support to fixed-line carriers. This extension is commonly called fixed mobile convergence (FMC). IMS/FMC allows subscribers to access any network (wireless or fixed) from any device (computer, PDA, or cell phone) and be able to move seamlessly from one network to another.

Figure 20 shows, at a high level, a converged IMS network that manages and controls the movement of subscribers between fixed and wireless networks.

Figure 20: A Simplified IMS Converged Network (Service Focus)



By itself, IMS does not specify new services; rather, it provides a framework for network operators to build and launch their services regardless of access method. The IMS architecture simplifies network operations and allows providers to focus on service introduction and business opportunities. For example, an IMS architecture could allow fixed and mobile users to communicate using voice, video, chat, and online gaming, and to take advantage of functionality such as Push-to-Talk over Cellular (PoC; the ability to quickly arrange meetings through a walkie-talkie mechanism), instant messaging, and presence (whether and how a subscriber is available, and how the subscriber wants to be contacted).

IMS and ETSI References

For more information about IMS and TISPAN, consult the following specifications:

- ETSI ES 283 026 V0.0.7 (2005-10) *Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); Resource and Admission Control; Protocol for QoS reservation information exchange between the Service Policy Decision Function (SPDF) and the Access-Resource and Admission Control Function (A-RACF) in the Resource and Protocol specification.*

- ETSI TS 183 017 V.0.0.8 (2005-10) *Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); Resource and Admission Control: DIAMETER protocol for session based policy set-up information exchange between the Application Function (AF) and the Service Policy Decision Function (SPDF); Protocol specification.*

- ETSI ES 283 034 V0.0.5 (2005-10) *Telecommunications and Internet converged Services and Protocols for Advanced Networks (TISPAN); Network Attachment Sub-System (NASS); e4 interface based on the DIAMETER protocol.*

Abbreviations

Table 10 identifies abbreviations used in the IMS and ETSI-TISPAN environments.

Table 10: Abbreviations in the IMS and ETSI-TISPAN Environments

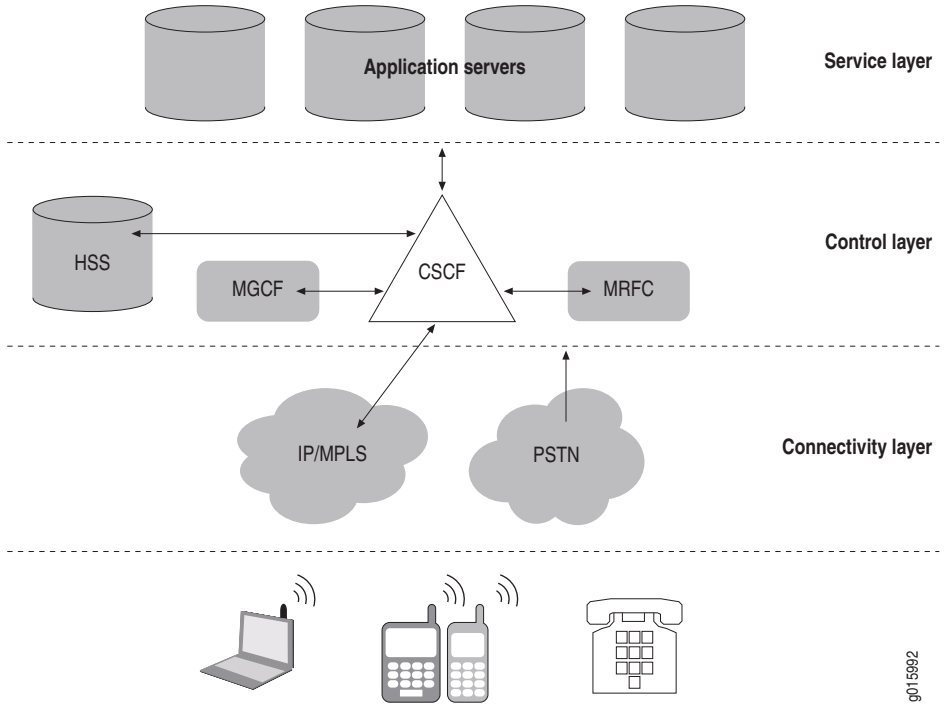
Abbreviation	Description
3GPP	3rd Generation Partnership Project, which developed the IMS specifications.
A-RACF	Access-resource and admission control function. Provides admission control and network policy assembly.
AVP	Attribute value pair
BGF	Border gateway function
ETSI	European Telecommunications Standards Institute
FMC	Fixed mobile convergence
IMS	IP multimedia subsystem
NGN	Next-generation network
RACS	Resource and admission control subsystem. Consists of the A-RACF and the SPDF.
RCEF	Resource control enforcement function
SPDF	Service policy decision function. The SPDF coordinates the resource reservations requests that it receives from the application function.
TISPAN	Telecommunications and Internet Converged Services and Protocols for Advanced Networks

IMS Layers

The IMS specifications define functions to handle the signaling and subscriber traffic for multimedia applications. The functions are separated into logical layers, and many of the specified functions often reside in a single platform. Vendors have the flexibility to implement IMS functions in consolidated ways, and it is natural that platforms such as softswitches will combine many logically separate IMS call-processing functions, and that routers will take on some of the session-enforcement and gateway functionality in IMS.

The three layers are the service layer, the control layer, and the transport layer. Figure 21 shows a high-level view of the IMS architecture.

Figure 21: High-Level View of the IMS Architecture



- Service layer—Hosts application and content services, including application servers and Web servers. It also includes generic service enablers that manage service elements such as user groups and presence. These service elements connect to subscribers through the control plane. The application layer supports most of the multimedia applications or application enablers, such as presence and location of the subscriber.

- Control layer—Makes the policy decisions that are enforced in the transport layer. This layer provides session control and management, and is responsible for setting up and taking down packet sessions. It also contains information about subscriber authentication, service authorization, and location.

- Connectivity layer—Supports the core network architecture of the General Packet Radio Service (GPRS), which consists of support nodes for data services. This layer is where routers, switches, firewalls, and optical transport reside, along with gateways that translate protocols between packet- and circuit-based traffic.

Signaling Protocol

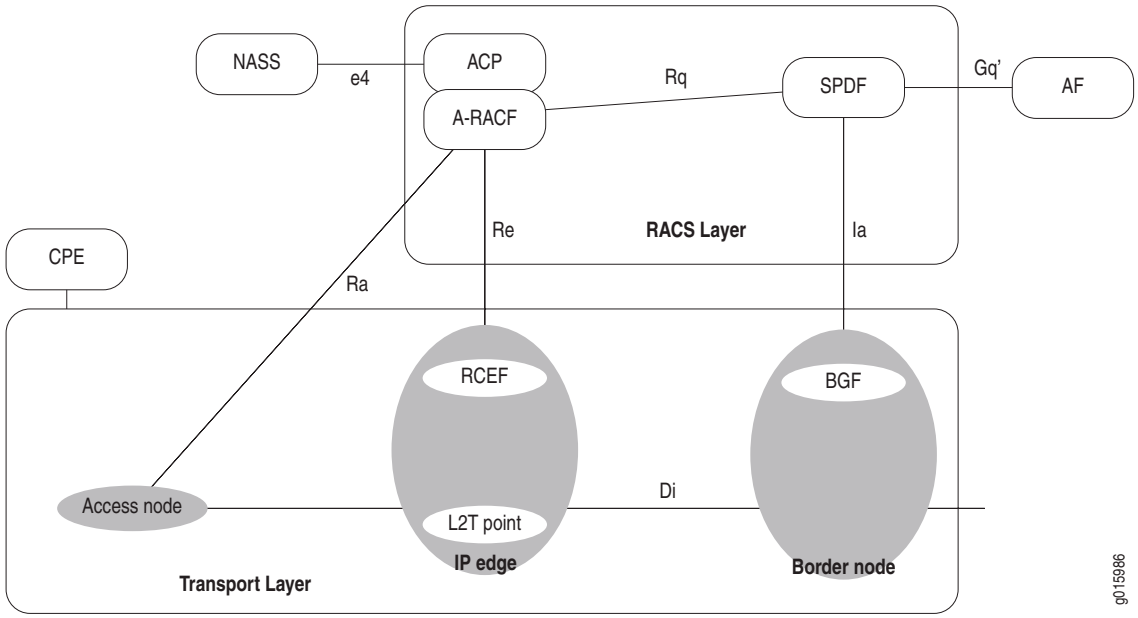
Session Initiation Protocol (SIP) is the main signaling protocol in IMS. SIP is the proposed standard for multimedia communication between subscribers interacting with voice, video, and instant messaging. In IMS, the use of SIP facilitates interconnectivity between fixed and mobile networks.

ETSI-TISPAN Architecture

TISPAN is an extension to the IMS architecture developed by ETSI to fit the specific requirements of fixed-line providers.

Figure 22 shows a high-level view of the TISPAN architecture.

Figure 22: High-Level View of the ETSI-TISPAN Architecture



RACS Layer

The RACS layer is the TISPAN next-generation network subsystem that is responsible for elements of policing control, including resource reservation and admission control in the access and aggregation networks. The RACS layer also includes support for NAT in the access, aggregation, and core networks required to support end-to-end application-initiated sessions.

The RACS provides policy-based transport control services to applications. These services enable applications to request and reserve transport resources from transport resources from the transport networks within the scope of the RACS.

Rq Interface

The Rq interface is the interface between the SPDF and the A-RACF. The SPDF issues requests for resources in the access network through the Rq interface. These requests indicate IP QoS characteristics. The A-RACF uses the IP QoS information to perform admission control and indicates to the SPDF through the Rq interface its admission control decisions.

SPDF

The SPDF is a functional element that coordinates the resource reservations requests that it receives from the application function (the application-level controller, such as a SIP server). The SPDF performs the following functions:

- Determines whether the request information received from the application function is consistent with the policy rules defined in the SPDF.
- Authorizes the requested resources for the application function session. The SPDF uses the request information received from the application function to calculate the proper authorization (that is, to authorize certain media components).
- Provides the location of the BGF and/or the A-RACF device, in accordance with the required transport capabilities.
- Requests resources of the A-RACF.
- Requests services from the BGF.
- Hides the details of the RACS and the core transport layer from the control architecture.
- Provides resource mediation by mapping requests from application functions toward an appropriate A-RACF and/or BGF.

A-RACF

The A-RACF is a functional element that provides admission control and network policy assembly.

For admission control, the A-RACF receives requests for QoS resources from the SPDF and uses the QoS information received to perform admission control. It then indicates to the SPDF whether or not a request for resources is granted.

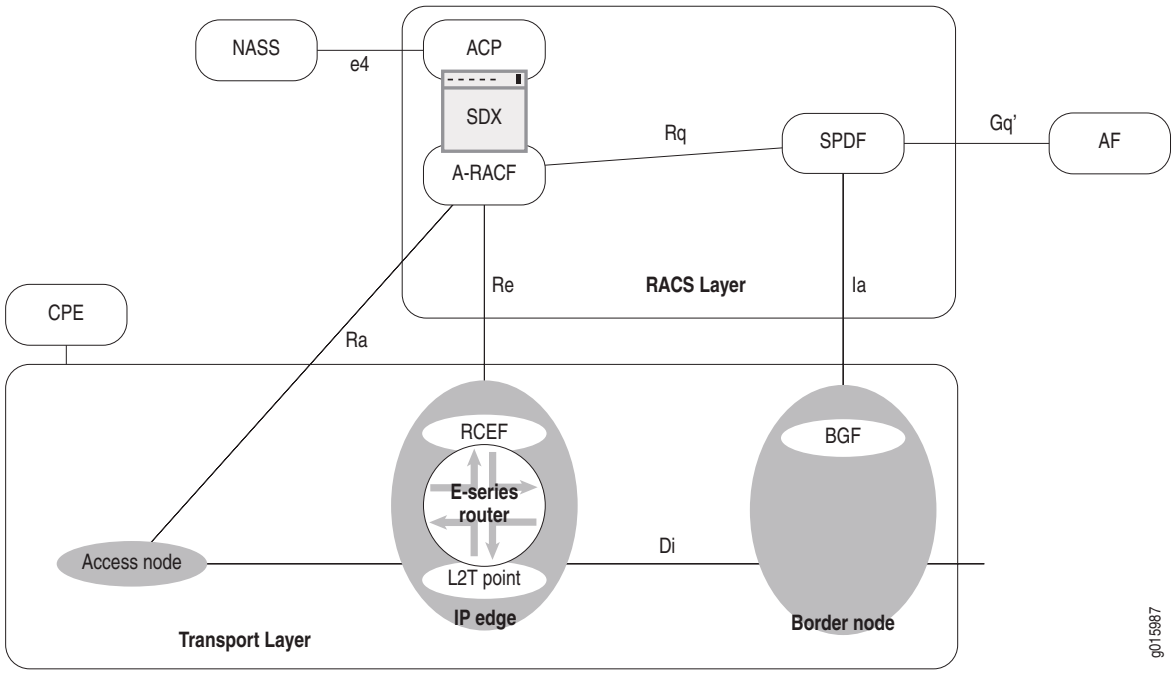
Access network policies are a set of rules that specify the policies that should be applied to an access line. For network policy assembly, the A-RACF:

- Ensures that requests from the SPDF match the access policies because multiple SPDFs can request resources from the A-RACF.
- Combines the requests from the SPDFs that have requested resources and ensures that the total of the requests match the capabilities of the access line.

SRC Software in the ETSI-TISPAN Architecture

Figure 23 shows the SRC software in the ETSI-TISPAN architecture.

Figure 23: SRC Software in the ETSI-TISPAN Architecture



The SAE provides the A-RACF functionality, and the SRC software provides a northbound Rq interface from the A-RACS to the SPDF. This interface is equivalent to the Rq interface defined in the ETSI-TISPAN release 1 architecture. It is a DIAMETER protocol–based interface that allows the SRC software to integrate with services found on the application layer of IMS.

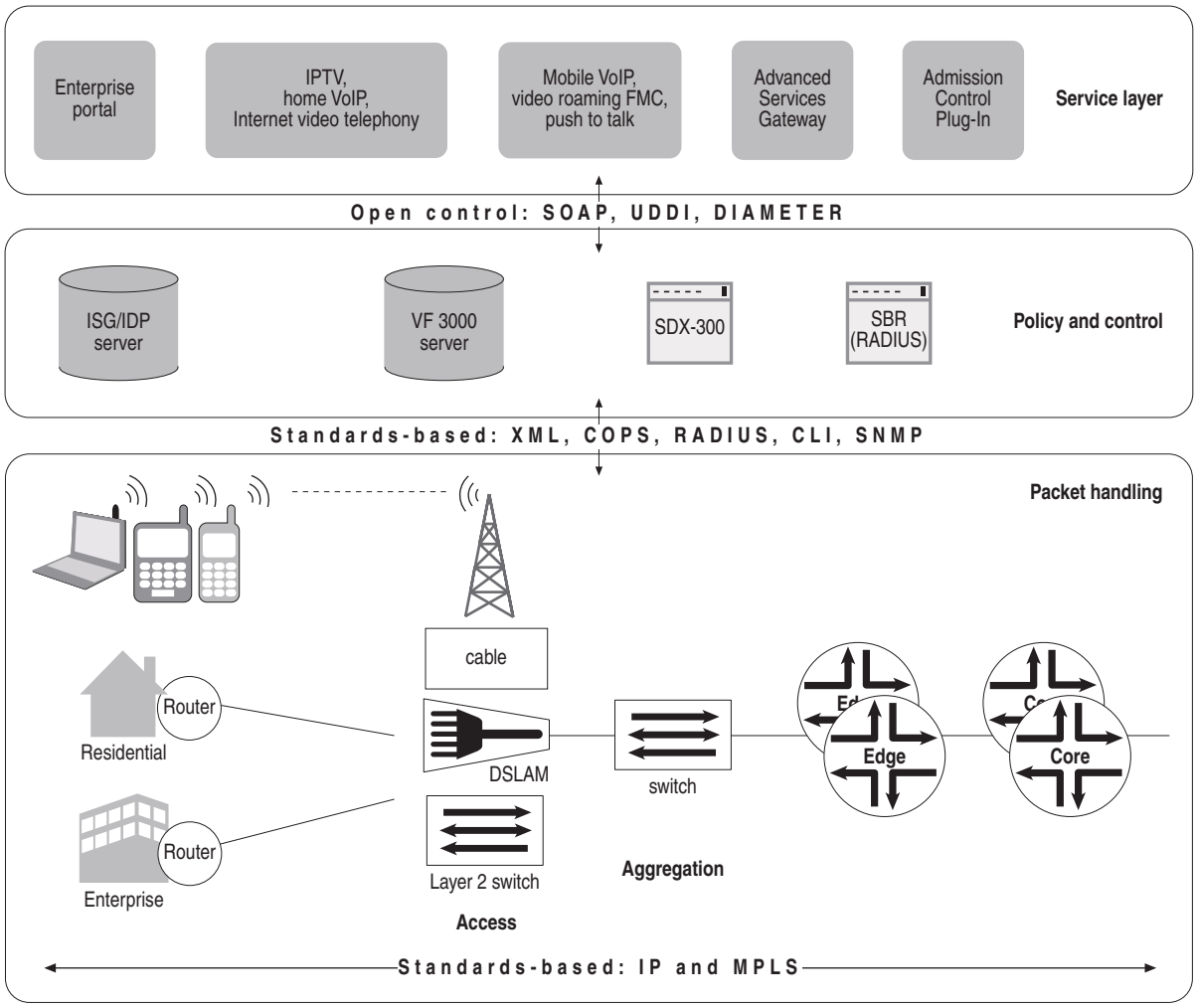
The SRC software uses its COPS and BEEP interfaces as the Re interface to Juniper Networks routers.

SRC Software in the IMS Environment

Figure 24 shows the Juniper Networks layered IMS architecture.

The northbound Rq interface of the policy and control layer allows integration with SRC applications, such as the portals, the Advanced Services Gateway, and the Admission Control Plug-In.

Figure 24: Juniper Networks IMS Architecture



Installing and Configuring the IMS Software

To install and configure the IMS software:

1. On the UNIX host where you will install the IMS software, log in as `root`.
2. Load SRC software disk 1 into the CD drive.
3. Install the UMCims package using the UNIX **pkgadd** tool.

   pkgadd -d /cdrom/cdrom0/SDX_DISK1/solaris10 UMCims
4. Follow the instructions on your screen to install the IMS software.

   The UMCims package is installed in the */opt/UMC/ims* folder.
5. Run the following command in the */opt/UMC/ims* folder.

   etc/config -a
6. Configure the local and remote DIAMETER peers in the */opt/UMC/ims/etc/config.properties* file.

   See *Configuration Fields for DIAMETER Peers* on page 187.
7. Configure logging destinations.

   See *Configuring Logging Destinations* on page 188.
8. Start the process to provide the A-RACF Rq interface.

   See *Starting the IMS Process to Provide the A-RACF Rq Interface* on page 191.

Configuration Fields for DIAMETER Peers

The properties in this section are in the */opt/UMC/ims/etc/config.properties* file.

local.address

- IP address of the local DIAMETER peer that is providing the A-RACF Rq interface.
- Value—IP address of the local host that is running A-RACF
- Default—127.0.0.1
- Property name—/ims/A-RACF/Rq/local.address

peer.1.remote.address

- IP address of the remote DIAMETER peer that is providing the SPDF Rq interface.
- Value—IP address of the host that is running SPDF.
- Default—127.0.0.1
- Property name—/ims/A-RACF/Rq/peer.1.remote.address

Configuring Logging Destinations

The properties in this section are in the */opt/UMC/ims/etc/config.properties* file. By default, the IMS has three logging destinations. To configure the logging destinations, modify the following parameters in the Logging section of the IMS config.properties file, where <loggerName> is a string that groups parameters for the logging destination.

For more information about logging, see *SDX Monitoring and Troubleshooting Guide, Chapter 4, Configuring Logging for SRC Components on a Solaris Platform*.

Logger.<loggerName>.class

- Specifies the type of logging.
- Value
  - file—Event messages are written to a text file.
  - stream—Event messages are written to stderr or stdout output.
  - syslog—Event messages are written to system log (syslog) facilities.

  If you do not fill in this field, the logging destination is disabled, and no logging is performed.
- Default—file

Logger.<loggerName>.filter

- Specifies the type of messages that this log file contains.
- Value—Filter definition. If you do not fill in this field, filtering is disabled.

  For more information about defining filters, see *Categories and Severity Levels for Event Messages* in the *SDX Monitoring and Troubleshooting Guide, Chapter 2, Configuring Logging for SRC Components*.
- Default
  - For Logger.log1.filter—/debug-
  - For Logger.log2.filter—/info-
  - For Logger.log3.filter—/error-

Logger.<loggerName>.filename

- Path of the file that contains the current logs for file-based logging.
- Value—Pathname
- Default
  - For Logger.log1.filename—*var/log/ims-a-racf-rq-debug.log*
  - For Logger.log2.filename—*var/log/ims-a-racf-rq-info.log*
  - For Logger.log3.filename—*var/log/ims-a-racf-rq-error.log*

Logger.<loggerName>.maxsize

- Maximum size of the log file for file-based logging.
- Value—Number of kilobytes in the range 0–4294967295
- Guidelines—Do not set the maximum file size to a value greater than the available disk space.
- Default—2000000000

Logger.<loggerName>.altfile

- Path of the alternate file. When the log file exceeds the maximum size specified by the Logger. < loggerName > .maxsize parameter, its contents are saved to this alternate file. If an alternate file already exists, it is overwritten.
- Value—Pathname
- Default
  - For Logger.log1.filename—*var/log/ims-a-racf-rq-debug.alt*
  - For Logger.log2.filename—*var/log/ims-a-racf-rq-info.alt*
  - For Logger.log3.filename—*var/log/ims-a-racf-rq-error.alt*

Logger.<loggerName>.stream

- Stream to use for stream-based logging.
- Value
  - stderr—Event messages are written to stderr output
  - stdout—Event messages are written to stdout output
- Default
  - For Logger.log1.stream—stdout
  - For Logger.log2.stream—stdout
  - For Logger.log3.stream—stderr

Logger.<loggerName>.hostname

- IP address or name of a host that collects event messages by means of a standard system logging daemon.
- Value—IP address or text string
- Default—localhost

Logger.<loggerName>.facility

- Specifies the type of system log in accordance with the system logging protocol.
- Value—Integer in the range 0–23; each integer corresponds to the standard number for a system logging client
- Default—No value

Logger.<loggerName>.format

- Specifies how the information in an event message is printed for syslog-based logging.
- Value—MessageFormat string as specified in

  http://java.sun.com/j2se/1.4.2/docs/api/java/text/MessageFormat.html

  The fields available for events are:

  - 0—Time and date of the event
  - 1—Name of the thread generating the event
  - 2—Text message of the event
  - 3—Category of the event
  - 4—Priority of the event
- Default—No value

Bootstrap Properties for IMS

The properties in this section are in the IMS bootstrap.properties file.

Config.java.naming.provider.url

- URL of the primary directory that stores configuration information.
- Value—ldap:// < host > : < portNumber >
  - < host > —IP address or name of host that supports the Web application
  - < portNumber > —Number of the TCP port
- Default—ldap://127.0.0.1:389/

Config.java.naming.security.credentials

- Password that the Web application server uses to authenticate and authorize gateway clients.
- Value— < password >
- Guidelines—The password can be encoded in base64 and not visible in plain text. To use an encoded value, use the format {BASE64} < encoded-value > .
- Default—conf

Config.java.naming.security.principal

- DN that contains the username that the Web application server uses to authenticate and authorize gateway clients.
- Value—DN of object that contains the username
- Default—*cn = conf, o = Operators, o = umc*

Config.net.juniper.smgt.lib.config.staticConfigDN

- Root of the static configuration properties.
- Value—DN of object that contains the username
- Default—*I = OnePop, I = NIC, ou = staticConfiguration, ou = configuration, o = Management, o = umc* (root of static configuration properties of sample data)

Config.net.juniper.smgt.lib.config.dynamicConfigDN

- Root of the dynamic configuration properties.
- Value—DN of object that contains the username
- Default—*ou = dynamicConfiguration, ou = configuration, o = Management, o = umc* (root of dynamic configuration properties of sample data)

Config.net.juniper.smgt.des.<propertySuffix>

- Set of properties that specify how IMS interacts with the directory.
- Values—See *SDX Getting Started Guide, Chapter 32, Distributing Directory Changes to SRC Components on a Solaris Platform*.
- Defaults—See *SDX Getting Started Guide, Chapter 32, Distributing Directory Changes to SRC Components on a Solaris Platform*.

Logger.file<propertySuffix>

- Set of properties that specify how IMS events are logged to files.
- Values—See *SDX Monitoring and Troubleshooting Guide, Chapter 4, Configuring Logging for SRC Components on a Solaris Platform*.
- Defaults—See *SDX Monitoring and Troubleshooting Guide, Chapter 4, Configuring Logging for SRC Components on a Solaris Platform*.

nic.<propertySuffix>

- Set of properties that configure the NIC proxy.
- Values—See *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 13, Configuring Applications to Communicate with an SAE*.
- Defaults—See *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 13, Configuring Applications to Communicate with an SAE*.

Starting the IMS Process to Provide the A-RACF Rq Interface

To start the IMS process to provide the A-RACF Rq interface:

1. On the IMS host, log in as `root` or as an authorized nonroot admin user.
2. Start the process from its installation directory.

   /opt/UMC/ims/etc/ims start

   The system responds with a start message.

Stopping the IMS Process to Provide the A-RACF Rq Interface

To stop the IMS process to provide the A-RACF Rq interface:

1. On the IMS host, log in as `root` or as an authorized nonroot admin user.
2. Stop the process from its installation directory.

   /opt/UMC/ims/etc/ims stop

   The system responds with a stop message.

Cleaning the IMS Log Files

To clean the IMS log files:

1. On the IMS host, log in as `root` or as an authorized nonroot admin user.
2. Enter the following command in the IMS installation directory.

   /opt/UMC/ims/etc/ims clean

Testing and Demonstrating the A-RACF Rq Interface

A sample SPDF that provides the Rq interface is included in the software for testing and demonstrating the A-RACF Rq. The SPDF Rq programs send activation requests, modification requests, and deactivation requests for the News service to the A-RACF Rq interface for various subscribers.

To run this program:

1. Run the following command in the */opt/UMC/ims* folder if you have not already done so.

   etc/config -a

2. Enter the following command:

   etc/SPDF-rq-sample appl [argument...]

   @param arguments
   args[0] address of the local peer, SPDF in this case.
   args[1] address of the remote peer, A-RACF in this case.
   args[2] number of seconds given the local peer to run. When
   the time is up the local will shutdown.
   args[3] number of seconds used in this range to wait before
   sending requests to the remote peer. For example a
   value of 25 means that from 0 to 25 seconds wait
   is inserted between each call (with and average
   delay of 12.5 seconds).
   args[4] number of subscribers to iterate over. The loop
   starts with the subscriber base address and is incremented
   by one in each step of the loop. For example, 100.
   args[5] subscriber base address. For example, 10.20.0.0.

Rq Interface Messaging

The following information provides a high-level description of the Rq interface and its messaging:

- A-RACF receives DIAMETER-messages from SPDF:
  - AA-Request for session initiation and for session modification
  - ST-Request for session termination
- In case of an AA-Request, the A-RACF verifies whether Session-Id AVP is new or already known. If new, a session is initiated; otherwise an existing session is modified.
- A-RACF gets the AF-Application-Id AVP within the Media-Component-Description AVPs to determine the service to be activated or modified.
- A-RACF retrieves the Framed-Ip-Address AVP within the Globally-Unique-Ip-Address AVPs to determine the IP address of the subscriber.
- A-RACF checks Flow-Status AVP. If the value is disabled, the session is reserved. If the value is enabled, the session is committed.
- The A-RACF reads the remaining DIAMETER AVPs and maps them according to the SAE external interface requirements.
- The A-RACF requests the required resources from the RCEF via the Re interface.
- A-RACF acknowledges the AA-Request with an AA-Answer back to the SPDF.

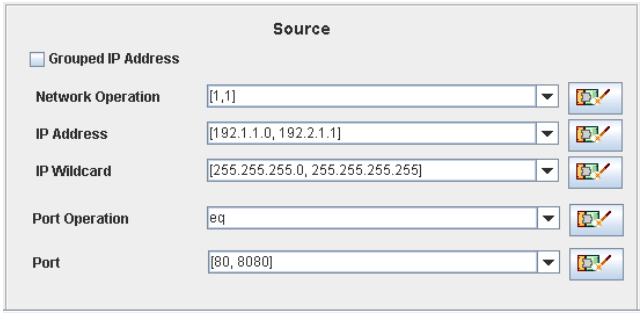
Configuring Policies for IMS

For IMS environments, you can configure JUNOSe policies. When you configure classify-traffic conditions, you can set up the software so that the SAE expands into multiple classifiers before it installs the policy on the router. If you enter a comma-separated list of values in the source and destination network (IP address, mask, and IP operation) or port fields (for port-related protocols), the software creates a classifier for each possible combination of address and port. Note that the software does not expand classifiers for values that are entered as a range.

For example, the source configuration in the classify-traffic condition in Figure 25 would cause the condition to be expanded into four classifiers that have the following combination of source addresses and source ports:

```
192.1.1.0/255.255.255.0 eq 8
192.1.1.0/255.255.255.0 eq 8080
192.2.1.1/255.255.255.0 eq 8
192.2.1.1/255.255.255.0 eq 80
```

Figure 25: Classify-Traffic Condition Example for Expanded Classifiers



Enabling Expansion of JUNOSe Classify-Traffic Conditions

To use SDX Configuration Editor to enable the expansion of JUNOSe classify-traffic conditions:

1. In the navigation pane, select a configuration file for the SAE that you want to configure.

2. Select the **Miscellaneous** tab, and expand the **Policy Management Configuration** section.

3. Edit or accept the default value.

   See *Enable JUNOSe Classifier Expansion Field* on page 195.

4. Select **File > Save**.

5. Right-click the configuration file, and select **SDX System Configuration > Export to LDAP Directory**.

Enable JUNOSe Classifier Expansion Field

In SDX Configuration Editor, you can edit the following field in the Policy Management Configuration section of the Miscellaneous pane in an SAE configuration file.

Enable JUNOSe Classifier Expansion

- Specifies whether or not the SAE expands the JUNOSe classify-traffic conditions into multiple classifiers before it installs the policy on the router.
- Value—Yes or No
- Guidelines—Because classifier expansion uses processing resources when the policy is created, you should set this property to true only if you are going to use the feature.
- Default—No

Chapter 19

Providing Prepaid Services

Prepaid service applications can be easily integrated with the SRC software. We provide the prepaid services demonstration application (referred to hereafter as the prepaid services demo) to illustrate the concept.

This chapter contains the following sections:

- Overview of Prepaid Services Demo on page 197
- Installing and Configuring the Prepaid Services Demo on page 199
- Managing Prepaid Accounts on page 203

Overview of Prepaid Services Demo

Prepaid service applications assume that users (subscribers) are registered with a subscriber management system, so that they can log in to the network and be authenticated. By default a subscriber has no access to the network. All attempts to access the Internet are intercepted and captured by the Service Selection Portal.

Subscribers pay for the service in advance of use by purchasing an access card that has a valid account number and expiration date. The subscriber then can activate the prepaid service, which might be, for example, access to the Internet or access to a gaming server. At activation, the portal prompts the subscriber for the account number and validates the access. The portal grants access if appropriate, and charging starts as soon as the portal grants access.

To integrate prepaid service applications, the prepaid services demo consists of the following two components:

- The prepaid account server is a Solaris package, UMCppdemo, located in the *solaris* directory on the SRC application library CD.
- The Prepaid Account Administration application is a standard Web application archive (WAR file) located on the SRC application library CD, */webapp/accountAdmin.war*. You must install this component in your application server, such as JBoss, and configure the object reference for the account server.

The prepaid services demo supports two types of prepaid service applications, time based and volume based. Both types are limited in the demo to a single service being concurrently active per prepaid account. The account server maintains the accounts.

Account Server

The account server is the central data repository for the prepaid services demo. It maintains the different accounts and provides access for the other SRC components. The account server is a CORBA server with a data storage backend. In the prepaid services demo, data is stored on the local file system; in a real application you should use a relational database management system (RDBMS) for data storage.

The account server employs the SAE plug-in interface. The server publishes an object reference to a standard COS naming service or to a file in the local file system. It uses the managed accounts to authorize access to prepaid services and updates the accounts based on actual usage.

The model assumes that subscribers can log in and be authenticated. By default, all attempts to access the Internet are intercepted and captured by the portal. Subscribers pay for the service in advance of use by purchasing a valid access card. The subscriber then can activate the prepaid service. At activation, the portal prompts the subscriber for the account number and validates the access. The portal grants access if the account exists, has not expired, is not locked, and has a remaining balance (time or volume) greater than 0. When it grants access in response to a request, the account server locks the account against concurrent access. Charging starts as soon as the portal grants access.

Time-Based Services

Time-based services are sold by access time. These services have no limits placed on the data transmitted. For example, voice long distance service accounts are measured in connection time.

When it authorizes a time-based prepaid account, the account server sets the session timeout based on the current balance of the account. The account server locks the account when the session start is signaled. When the session stop is signaled, the account server updates the account based on the session time and unlocks it.

When the service stops (because the subscriber stops service on the portal, the subscriber logs out, or the session timeout expires), the account is unlocked, and its time balance is decreased by the session time.

Volume-Based Services

Volume-based services are sold based on upload or download data volume. For the demo application, volume is defined as the sum of the upload and download volumes. A real implementation might distinguish between the two for accounting purposes.

When it authorizes a volume-based prepaid service, the account server sets an interim update interval according to the following formula:

$$\text{interim update interval} = \frac{\text{remaining volume in account}}{\text{maximum bandwidth available to subscriber}}$$

The maximum bandwidth is the greater of the two plug-in attributes upstreamBandwidth and downstreamBandwidth. If you do not specify values for these attributes, then the maximum bandwidth defaults to 1 Mbps.

When the session start is signaled, the account server locks the account.

When an interim update is signaled, the account server updates the interim update interval based on the account balance and the current consumption. It compares the volume used so far in the session with the remaining volume. If the session volume is greater than the remaining volume, the account server sets the session timeout to zero to stop the session.

If the session volume is less than the remaining volume, the interim update interval is recalculated, and no further action takes place until the end of that interval.

The interim update interval must be larger than a specified minimum value; the demo application employs a minimum of 5 minutes. This feature enables accounts to be overdrawn by an amount equal to the maximum bandwidth times the minimum time.

When the session stop is signaled, the account server updates the account based on the volume counters and unlocks it.

Installing and Configuring the Prepaid Services Demo

You must install and configure both the account server and the Prepaid Account Administration application. Additionally, you must configure the prepaid plug-in on the SAE and create and configure the service(s) that will use the prepaid accounts.

Installing the Account Server

You must manually install the UMCppdemo package to deploy the account server.

```
pkgadd -d /cdrom/cdrom0/solaris UMCppdemo
```

NOTE: The prepaid services demo is provided on the SRC application library CD.

For information about installing the prepaid services demo, see *SRC Application Library Guide, Chapter 1, Installing the SRC Applications*.

Configuring the Account Server

Before you start the account server for the first time, you must run a script to configure it. To configure the account server:

1. On the SAE host, log in as `root` or as an authorized nonroot admin user.

2. Launch the configuration script from the prepaid services demo installation directory.

   /opt/UMC/prepaid/etc/config

3. The configuration script prompts you for input and confirms your choices, as in the following example:

   ```
   Which naming prefix shall be used for publishing the objects?
   [demo/accountServer] [?,q]
   demo/accountServer
   Which naming server do you want to use? [] [?,q]
   corbaname::localhost
   Which file name prefix shall be used for publishing the objects? [] [?,q]
   /var/tmp/accountServer
   Which user-id shall be running the account server? [nobody] [?,q]
   nobody
   COSName:    "demo/accountServer"
   NameServer: "corbaname::localhost"
   IORFile:    "/var/tmp/accountServer"
   USERID:     "nobody"
   Is this correct? y
   ```

4. The script configures the account server according to your responses.

Publishing the Object References

The sample configuration presented above configures the account server to publish the object references to a COS naming service and to a local file. Depending on your needs, you might want to choose only one or the other method.

NOTE: The account server and the Prepaid Account Administration application must run on the same host for the local file feature to work. If you install these components on multiple hosts, you must configure the account server to publish the object references to a COS naming service.

When you publish the objects to a COS naming service, you specify the prefix of the published name, such as demo/accountServer, and the URL of the name server, such as corbaname::localhost. In this case the account server publishes the object reference of the plug-in to the URL corbaname::localhost#demo/accountServer.plugin. The account server publishes the object reference of the account manager to the URL corbaname::localhost#demo/accountServer.acctMgr.

The local file is specified by the path and prefix of the filename, */var/tmp/accountServer*. In this case the account server publishes the object reference of the account manager to */var/tmp/accountServer.acctMg*r and the object reference of the prepaid plug-in to */var/tmp/accountServer.plugin*.

Manual Configuration

Although the configuration script is sufficient to configure the account server for most purposes, you can also configure the server by using the command line.

- To publish the object references into a local file, specify the path and prefix of the filename:

  ```
  #accountServer -f <fileNamePrefix>
  ```

- To publish the object references to a COS naming service, specify the prefix of the published name:

  ```
  #accountServer -c <namePrefix>
  ```

- The COS naming server is taken from the initial references. You can do one of the following:

  - Globally configure omniORB in the file */etc/omniORB.cfg*
  - Specify the following option when you configure the account server:

    ```
    -ORBInitRef NameService=corbaname::nameServerHostname
    ```

  For example, to publish the object references to a COS naming server running on server ns.domain.com, configure the account server as follows:

  ```
  #accountServer -ORBInitRef NameService=corbaname::ns.domain.com -c
  demo/accountServer
  ```

- If you start the account server as a root user, the account server switches the user ID to an unprivileged user after initialization. The default user ID is nobody. To override the default value, specify a different user:

  ```
  #accountServer -f /var/tmp/accountServer -u <username>
  ```

Starting the Account Server

To start the account server:

1. On the account server host, log in as `root` or as an authorized nonroot admin user.
2. Start the account server from the root directory.

   /etc/init.d/accountServer start

   The system responds with a start message.

Stopping the Account Server

To stop the account server:

1. On the account server host, log in as `root` or as an authorized nonroot admin user.

2. Stop the account server from the root directory.

   /etc/init.d/accountServer stop

   The system responds with a stop message.

Configuring the SAE for the Prepaid Plug-In

You configure the prepaid plug-in in the same way that you configure other SAE external plug-ins. For information about configuring SAE plug-ins, see:

- *SDX Subscribers and Subscriptions Guide, Chapter 9, Configuring Internal, External, and Synchronization Plug-Ins with the SRC CLI*
- *SDX Subscribers and Subscriptions Guide, Chapter 10, Overview of Configuring Plug-Ins for Solaris Platforms*

The properties for this plug-in are as follows.

Plugin.prepaid.objectref

- Specifies the reference of the plug-in object implemented by the account server.
- Value—Depends on the host and the configuration of the account server; that is, whether the object reference is published to a COS naming service or a local file
- Examples

  In the following example, the object reference has been published to a COS naming service running on the host ns.domain.com:

  Plugin.prepaid.objectref = corbaname::ns.domain.com#demo/accountServer.plugin

  In the following example, the object reference has been published to a local file on the host:

  Plugin.prepaid.objectref = file:/var/tmp/accountServer.plugin

Plugin.prepaid.attr

- Defines the attributes used by the plug-in.
- Value—Use only the following value:
  Plugin.prepaid.attr = PA_UID,PA_AUTH_USER_ID, PA_SESSION_TIME, PA_DOWNSTREAM_BANDWIDTH, PA_UPSTREAM_BANDWIDTH

Configuring the Prepaid Services

Each defined service that uses prepaid accounts must be configured to use the prepaid plug-in as its authorization and tracking plug-in. For example, suppose you have a GameMaster premium gaming service for which you want to use prepaid accounts. You must create this service with SDX Admin and enter the value "prepaid" into the Authorization Plugin and Tracking Plugin fields. See *SDX Services and Policies Guide, Chapter 2, Managing Services on a Solaris Platform* for more information about creating and configuring services with SDX Admin.

Deploying the Prepaid Account Administration Application

You must deploy the WAR file for the Prepaid Account Administration application in the Web application server.You can find this file, *accountAdmin.war*, in the folder *webapp* on the SRC application library CD. Refer to the documentation for your Web application server for information about deploying applications.

For example, to deploy the Prepaid Account Administration application inside JBoss, copy the file to the JBoss */server/default/deploy* directory.

```
cp /cdrom/cdrom0/webapp/accountAdmin.war
/opt/UMC/jboss/server/default/deploy
```

JBoss automatically starts the application when a new WAR file is copied into the deploy directory.

Configuring the Prepaid Account Administration Application

You must configure the Prepaid Account Administration application with the object reference of the account manager. Configure the object reference as a <context-param> in the *WEB-INF/web.xml* file from the *accountAdmin.war* file. The parameter name is acctMgr, and the value is a CORBA URL of the account manager object reference, as in the following example:

```
<context-param>
  <param-name>acctMgr</param-name>
  <param-value>corbaname::ns.domain.com#demo/accountServer.acctMgr
</param-value>
</context-param>
```

Managing Prepaid Accounts

Use the Prepaid Account Administration application to manage prepaid accounts.

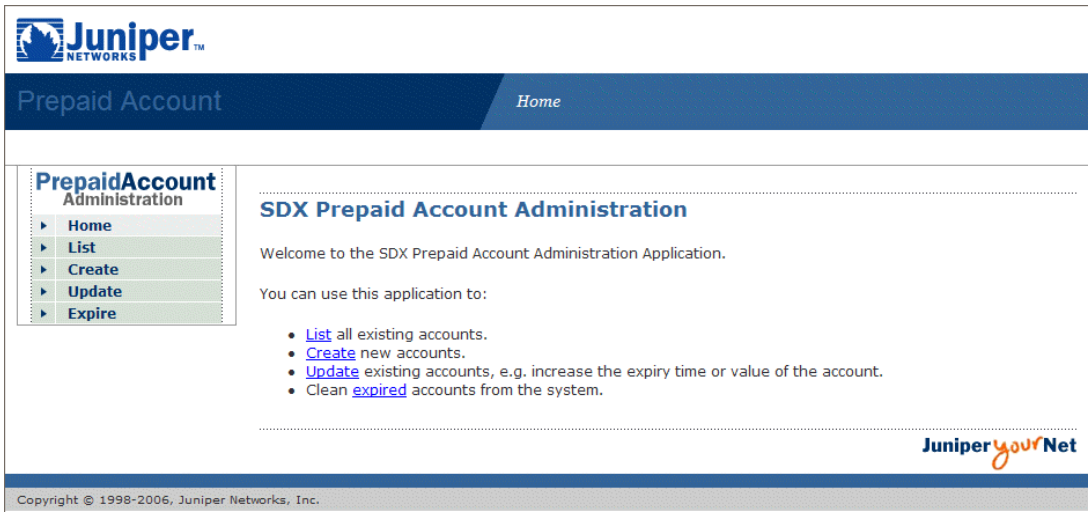
Accessing the Prepaid Account Administration Application

To access the Prepaid Account Administration application, enter the following in your Web browser:

```
http://<host>:8080/accountAdmin
```

where <host> is the name or IP address of the workstation on which you installed the Prepaid Account Administration application.

The Prepaid Account Home page appears.



Administering Accounts

On the Prepaid Account Home page, you can select to list, create, update, or clear accounts.

- Listing an Account—You can display summary information for accounts. You can also display the state of an account selected by its account number.

- Creating an Account—On the Create Accounts page, you can create multiple similar accounts simultaneously, but all must have the same type (time or volume), balance, and expiration (expiry) date.

  Specify the expiration date in the format YYYYMMDD. You can optionally specify an expiration time after the date in the format HHmm. If you do not specify a time, the account expires at midnight on the specified date. For example, to set the expiration for July 21, 2004 at 08:35 a.m., specify the following in the Expiry Date field:

  200407210835

  After completing the account fields, click OK to create the account(s).

- Updating an Account—On the Update Accounts page, you can manually credit an account, extend its expiration date, or unlock it.

- Clearing an Account—On the Clear Accounts page, you can delete expired accounts.

Index

A

accessing
  Prepaid Account Administration application 203
account server, prepaid services demo 198
  manual configuration 201
  publishing object references 200
  script configuration 200
  starting 201
  stopping 202
accounts
  administering with Prepaid Account Administration application 204
address pools
  assigned IP subscribers
    configuring 127, 140
address pools. *See* IP address pools 79
application manager
  role, in PCMM environment 38
applications
  SRC on CD xiii
assigned IP subscribers
  PCMM network 48, 79, 85
    address pools 127, 140
    IP address pools 48
  setting timeouts 36
  voice over IP 35
audience for documentation xi

C

cable modem termination system. *See* CMTS devices
classify-traffic condition
  expanded classifiers
    configuring 194
client type 1, PCMM 40
client type 2, PCMM 41
CMTS devices
  adding objects to directory
    SDX Admin 81
    SRC CLI 77
  adding virtual router objects to directory
    SDX Admin 83
    SRC CLI 78
  configuration statements 77, 78
  role 38
CMTS locator
  monitoring
    C-Web interface 155
    SRC CLI 149
configuring
  SAE for SRC-ACP 171
conventions defined
  icons xii
  text xii
CORBA (Common Object Request Broker Architecture)
  reference for SAE
    PCMM 86
custom RADIUS authentication plug-in 27
customer support xv

D

Data over Cable Service Interface Specifications. *See* DOCSIS protocol
demonstration application, prepaid services 197
deploying
  Prepaid Account Administration application 203
DOCSIS protocol 39
documentation set, SRC. *See* SRC documentation set
domains
  IP service edge 43
  IP subscriber edge 43
  radio frequency 43
dynamic RADIUS authorization requests
  RADIUS packets, defining 166

E

end-to-end services 43
event notification
  DHCP server 167
  IP address manager 167
  PCMM network 167
  RADIUS server 167
event notification, PCMM network
  configuration statements 58
  description 49
  properties, configuring
    SDX Configuration Editor 70
    SRC CLI 58
events, publishing 173

expanded classifiers
configuring ... 194

F

flexible RADIUS authentication plug-ins
configuring ... 27

I

icons defined, notice ... xii
interim update interval ... 199
IOR
managing SAE
PCMM ... 86
IP address managers
logging in subscribers
event notification method ... 167
IP address pools
assigned IP subscribers ... 48
assigned IP subscribers, configuring
SDX Admin ... 85
SRC CLI ... 79
local address pools, configuring
SDX Admin ... 85
SRC CLI ... 79
static pools, configuring
SDX Admin ... 86
SRC CLI ... 79
syntax ... 85, 140
IP Security. *See* IPSec
IPSec
between SAE and other applications ... 91–102
changing configuration ... 102
configuration prerequisites for SAE ... 94
configuring for SAE ... 93, 95
keys
automatic key management ... 93
overview ... 92
supported ... 92
overview ... 91
testing connection ... 103
IPTV application
configuring ... 169
installing ... 168

J

JPS (Juniper Policy Server)
application manager-to-policy server interface, configuring ... 115
application manager-to-policy server interface, monitoring
C-Web interface ... 152, 153
SRC CLI ... 148
architecture ... 106
CMTS devices, monitoring
C-Web interface ... 154
CMTS locator, monitoring
C-Web interface ... 155
SRC CLI ... 149
installing on Solaris platforms ... 133
JPS state, monitoring ... 148
local configuration, applying on Solaris platforms ... 134
logging, configuring ... 114
logging, modifying ... 114
message flows, monitoring
C-Web interface ... 156
SRC CLI ... 149
message handler, monitoring
C-Web interface ... 155
SRC CLI ... 149
monitoring
C-Web interface ... 151
Solaris platforms ... 136, 145
SRC CLI ... 131, 147
NTP configuration
applying on Solaris platforms ... 134
modifying local time on Solaris platforms ... 134
operational status ... 131, 136
overview ... 105
policy server-to-CMTS interface, configuring ... 121
policy server-to-CMTS interface, monitoring
C-Web interface ... 153, 154
SRC CLI ... 148
policy server-to-RKS interface, configuring ... 117
policy server-to-RKS interface, monitoring
C-Web interface ... 157
SRC CLI ... 148
server process, monitoring
C-Web interface ... 151
SRC CLI ... 147
starting
Solaris platforms ... 135
SRC CLI ... 131
stopping
Solaris platforms ... 136
SRC CLI ... 131
subscriber address mappings, configuring ... 124
subscriber configuration, modifying ... 124
Juniper Policy Server. *See* JPS

L

login process
assigned IP subscribers, PCMM ... 48
event notification method, PCMM ... 50

M

manuals, SRC
comments ... xv

N

NIC (network information collector)
IP address pool ... 140
IP address pool, configuring
SDX Admin ... 85
SRC CLI ... 79
local IP address pools ... 89
NIC configuration scenarios
OnePopDynamicIp ... 89, 90
PCMM environment, in ... 89, 90
notice icons defined ... xii

O

object references, publishing ... 200
objectives of guide ... xi

P

packet mirroring, configuring ... 160
PacketCable Multimedia. *See* PCMM
PCMM (PacketCable Multimedia)
application manager, role ... 38
client type 1 ... 40
client type 2 ... 41
CMTS device, role ... 38
configuring SAE
SDX Configuration Editor ... 65
SRC CLI ... 53
creating sessions ... 48
description ... 38–42
DOCSIS protocol ... 39
end-to-end QoS architecture ... 43
end-to-end services ... 43
integrating SRC software ... 37
IP service edge domain ... 43
IP subscriber edge domain ... 43
logging in subscribers
assigned IP method ... 48
event notification method ... 49
overview ... 48
NIC configuration for ... 89
overview ... 37
policy server, role ... 38
provisioning end-to-end services ... 45
record-keeping server ... 38
RF domain ... 37
SAE ... 47
SAE communities ... 51
service flows ... 39
session store ... 52
single-phase resource reservation model ... 40, 41
SRC software in
description ... 42
traffic profiles ... 43
videoconferencing example ... 45
video-on-demand example ... 46
PCMM device driver
configuration statements ... 54
configuring
SDX Configuration Editor ... 66
SRC CLI ... 54
PCMM record-keeping server plug-in
configuration statements ... 60
configuring
SDX Configuration Editor ... 71
SRC CLI ... 60
description ... 52
plug-ins
PCMM record-keeping server plug-in ... 52
prepaid plug-in ... 202
policy servers
adding application manager groups
SDX Admin ... 138
SRC CLI ... 126
adding objects to directory
SDX Admin ... 142
SRC CLI ... 128
role, in PCMM architecture ... 38
specifying application managers
SDX Admin ... 138
SRC CLI ... 126
specifying SAE communities
SDX Admin ... 138
SRC CLI ... 126
Policy Web Admin
connecting to directory ... 22
launching ... 21
querying directory ... 23
searching for QoS Policy data ... 20
Prepaid Account Administration application ... 197, 203
accessing ... 203
administering accounts ... 204
configuring ... 203
deploying WAR file ... 203
prepaid plug-in configuration ... 202
prepaid services demonstration application ... 197
account server ... 198
components ... 197
configuring ... 199
configuring prepaid services ... 203
installing ... 199
interim update interval ... 199
SAE configuration ... 202

time-based prepaid services 198
volume-based prepaid services 198
publishing events 173

Q

QoS (quality of service)
PCMM environments
description 37
end-to-end QoS architecture 43
extending to service edge domain 44
extending to subscriber edge domain 44
searching for policies in directory 17, 20
QoS profiles, JUNOSe routers
how tracking works 2
managing dynamically 2–7
updating directory, using
qosProfilePublish 14
SDX Admin 14
QoS profile-tracking plug-in
configuring
SDX Configuration Editor 10
description 2
quality of service. See QoS

R

RADIUS
vendor-specific attributes for wireless ISP roaming 27
record-keeping server. *See* RKS
release notes xv
RKS (record-keeping server)
peers, configuration statements 59
peers, configuring in plug-ins
SDX Configuration Editor 74
SRC CLI 59
plug-in 52
plug-in, configuration statements 63
plug-in, configuring
SDX Configuration Editor 71
SRC CLI 60
role in PCMM environment 38
roaming wireless environment 25–32

S

SAE (service activation engine)
configuring as an application manager
Solaris platforms 137
SRC CLI 125
configuring for SRC-ACP 171
idle timeout 32
PCMM environment 47
redundancy. *See* SAE communities
SAE (service activation engine), configuring
IPSec 93
community manager
SDX Configuration Editor 69
SRC CLI 57
event notification API properties
SDX Configuration Editor 70
SRC CLI 58
IOR, PCMM 86
PCMM device driver
SDX Configuration Editor 66
SRC CLI 54
SAE communities
configuration overview
SRC CLI 57
configuration statements 57
configuring
SDX Configuration Editor 69
configuring manager
SDX Configuration Editor 69
SRC CLI 57
defining members
SDX Admin 84
SRC CLI 79
description 51
service flows 39
services
configuring prepaid 203
time-based 198
voice over IP (VoIP) 33
volume-based 198
session store
in PCMM environment 52
single phase resource reservation model, PCMM 40, 41
Solaris packages
UMCppdemo 197
SRC documentation set
comments xv
obtaining xv
SRC documentation CD xiii
SRC software distribution xv
SRC-ACP (SRC Admission Control Plug-In)
event publishers, configuring 173
SAE, configuring for 171
subscriber
wireless environment 26
support, requesting xv

T

technical support, requesting xv
text conventions defined xii
traffic profiles, PCMM 43

U

UMCppdemo Solaris package 197

W

Web Admin applications 197, 203
wireless environment 25–32