SRC-PE Software

Network Guide: SAE, Juniper Networks Routers, NIC, and SRC-ACP

Release 2.0.x

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

Part Number:530-022125-01, Revision 01

Products made or sold by Juniper Networks (including the ERX-310, ERX-705, ERX-710, ERX-1410, ERX-1440, M5, M7i, M10, M10i, M20, M40, M40e, M160, M320, and T320 routers, T640 routing node, and the JUNOS, JUNOSe, and SDX-300 software) or components thereof might be covered by one or more of the following patents that are owned by or licensed to Juniper Networks: U.S. Patent Nos. 5,473,599, 5,905,725, 5,909,440, 6,192,051, 6,333,650, 6,359,479, 6,406,312, 6,429,706, 6,459,579, 6,493,347, 6,538,518, 6,538,899, 6,552,918, 6,567,902, 6,578,186, and 6,590,785.

SRC-PE Software Network Guide: SAE, Juniper Networks Routers, NIC, and SRC-ACP, Release 2.0.x
Writing: Linda Creed, Justine Kangas, Betty Lew, Helen Shaw
Editing: Fran Mues
Illustration: Nathaniel Woodward
Cover Design: Edmonds Design

Revision History
11 October 2007—Revision 1

The information in this document is current as of the date listed in the revision history.

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

Part 2 Using Juniper Networks Routers in the SRC Network

Chapter 5 Using JUNOSe Routers in the SRC Network with the SRC CLI 45

Chapter 6 Using JUNOSe Routers in the SRC Network with a Solaris Platform 67

Chapter 8 Using JUNOS Routing Platforms in the SRC Network with a Solaris Platform 115

Part 3 Locating Subscriber Management Information

Chapter 18 Customizing a NIC Configuration 231

Chapter 19 NIC Configuration Scenarios 237

Part 4 Providing Admission Control with SRC-ACP

About This Guide

This preface provides the following guidelines for using the *SRC-PE Software Network Guide: SAE, Juniper Networks Routers, and NIC*.

- Objectives on page xix
- Audience on page xix
- Documentation Conventions on page xx
- Related Juniper Networks Documentation on page xxi
- Obtaining Documentation on page xxiii
- Documentation Feedback on page xxiii
- Requesting Support on page xxiv

Objectives

This guide provides an overview of the service activation engine (SAE) and describes how to use it. The guide also describes how to use Juniper Networks routers in the Session and Resource Control (SRC) network, how to use the NIC to locate subscriber management information, and how to use the SRC Admission Control Plug-In (SRC-ACP) application to provide admission control.

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;
    }
```

Table 2: Text Conventions (continued)

Convention	Description	Examples
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>.
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.*

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	
C2000 and C4000 Hardware Guide	Describes the hardware platforms and how to install, maintain, replace, and troubleshoot them. The guide also includes specifications.
C2000 and C4000 Quick Start Guide	Describes how to get the C-series Controller up and running quickly. Intended for experienced installers who want to expedite the installation process.
SRC-PE Getting Started Guide	Describes the SRC software, how to set up an initial software configuration, and how to upgrade the SRC software. It also explains how to manage a C-series Controller. The guide describes how to set up and start the SRC CLI and the C-Web interface, as well as other SRC configuration tools. It includes reference material for the SRC documentation.
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, NIC, and SRC-ACP	Describes how to use and configure the SAE, the NIC, and the SRC-ACP (Admission Control Plug-In) application. 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 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); and mirroring subscriber traffic on JUNOSe routers.
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 NETCONF API Guide	Describes how to use the NETCONF application programming interface (API) to configure or request information from the NETCONF server on a C-series Controller that runs the SRC software.
SRC-PE XML API Configuration Reference	Describes the tag elements in the SRC Extensible Markup Language (XML) application programming interface (API) that are equivalent to configuration statements in the SRC command-line interface (SRC CLI).
SRC-PE XML API Operational Reference	Describes the tag elements in the SRC Extensible Markup Language (XML) application programming interface (API) that are equivalent to operational commands in the SRC command-line interface (SRC CLI).

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

Document	Description
SRC-PE Comprehensive Index	Provides a complete index of the SRC guides, excluding the *C-series Hardware Guide*, the *SRC CLI Command Reference,* the *SRC-PE NETCONF API Guide,* the *SRC-PE XML API Configuration Reference,* and the *SRC-PE XML API Operational Reference.*
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, an application to provide threat mitigation, 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, and an application to control volume usage.
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. 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).

Part 1

Operating the SAE

Chapter 1

Overview of the SAE

This chapter gives an overview of the features of the SAE. Topics include:

- Role of the SAE on page 3
- Connections to Managed Devices on page 4
- SAE Plug-Ins on page 6
- Tracking and Controlling Subscriber and Service Sessions with SAE APIs on page 8
- SAE Accounting on page 10

Role of the SAE

The SAE is the core manager of the SRC network. It interacts with other systems, such as Juniper Networks routers, cable modem termination system (CMTS) devices, directories, Web application servers, and RADIUS servers, to retrieve and disseminate data in the SRC environment. The SAE authorizes, activates and deactivates, and tracks subscriber and service sessions. It also collects accounting information about subscribers and services.

The SAE makes decisions about the deployment of policies on JUNOSe routers and JUNOS routing platforms. When a subscriber’s IP interface comes up on the router, the SAE determines whether it manages the interface. If the interface is managed—or controlled—by the SAE, the SAE sends the subscriber’s default policy configuration to the router. These default policies define the subscriber’s initial network access. When the subscriber activates a value-added service, the SAE translates the service into lists of policies and sends them to the router.

The SAE also provides plug-ins and application programming interfaces (APIs) that extend the capabilities of the SRC software.

Connections to Managed Devices

This section describes the connections between the SAE and Juniper Networks routers, CMTS devices, and the Juniper Policy Server (JPS).

COPS Connection Between JUNOSe Routers and the SAE

The SAE and JUNOSe routers communicate using the Common Open Policy Service (COPS) protocol. The SAE supports two versions of COPS:

- COPS usage for policy provisioning (COPS-PR)
- COPS External Data Representation Standard (XDR) mode

The version of COPS that you use depends on the version of COPS that your JUNOSe router supports. When you set up your JUNOSe router to work with the SAE, you enable either COPS-PR mode or COPS XDR mode. There are no configuration differences on the SAE between COPS-PR and COPS XDR.

The following SRC features require the use of COPS-PR:

- Policy sharing on JUNOSe routers
- Multiple classify traffic conditions in policy lists

For more information, see one of the following:

- *Chapter 5, Using JUNOSe Routers in the SRC Network with the SRC CLI*
- *Chapter 6, Using JUNOSe Routers in the SRC Network with a Solaris Platform*.

Beep Connection Between JUNOS Routing Platforms and the SAE

The SAE interacts with a JUNOS software process, referred to as the SRC software process, on a JUNOS routing platform. The SAE and the SRC software process communicate using the Blocks Extensible Exchange Protocol (BEEP).

When a JUNOS routing platform that the SAE manages goes online, it initiates a BEEP session for the SAE. The SAE gets configuration information from the router, and then it builds and installs the policies that control the router’s behavior. If the policies are subsequently modified in the directory, the SAE builds a new configuration and reconfigures the interface on the JUNOS routing platform.

NOTE: The SAE manages interfaces on JUNOS routing platforms only when the interfaces are configured in the global configuration and the router sends added, changed, or deleted notifications to the SAE. Router administrators should not manually change the configuration of interfaces that the SAE is managing. If you manually change a configuration, you must remove the SAE from the system.

When there are configuration changes on the router, the router sends a notification to the SAE through the BEEP connection. The notification does not include the content of the configuration changes. When the SAE receives the notification, it uses its JUNOScript client to get the changed configuration from the router.

Interfaces that have been deleted from the router along with their associated objects (sessions, policies) remain on the router until state synchronization occurs.

For more information, see one of the following:

- *Chapter 7, Using JUNOS Routing Platforms in the SRC Network with the SRC CLI*
- *Chapter 8, Using JUNOS Routing Platforms in the SRC Network with a Solaris Platform*

COPS Connection Between CMTS Devices and the SAE

The SAE uses the COPS protocol as specified in the PacketCable Multimedia Specification PKT-SP-MM-I03-051221 to manage *PacketCable Multimedia Specification* (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.

For more information, see *SRC-PE Solutions Guide, Chapter 4, Providing Premium Services in a PCMM Environment*.

COPS Connection Between Juniper Policy Servers and the SAE

When the SAE is acting as an application manager in a PCMM environment, it connects to the JPS through an interface on the JPS. The JPS uses the COPS protocol as specified in the PacketCable Multimedia Specification PKT-SP-MM-I03-051221 for its interface connections. The JPS communicates with the application manager by using a COPS over TCP connection.

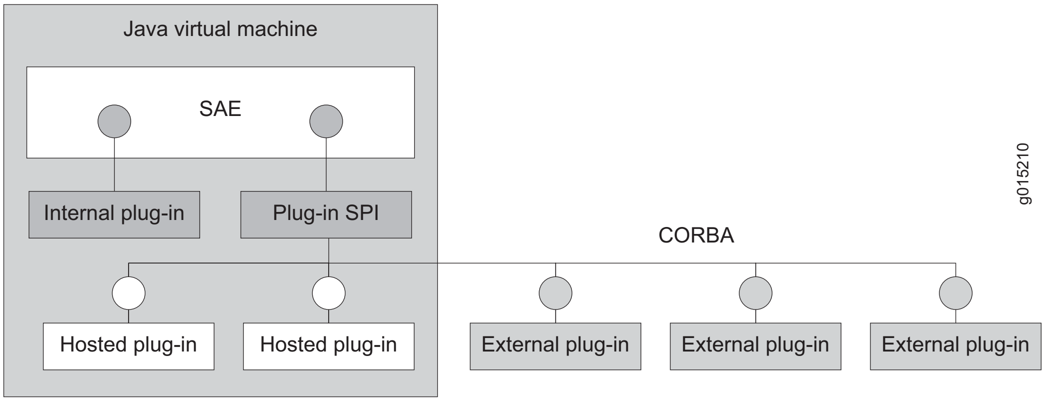
For more information, see *SRC-PE Solutions Guide, Chapter 9, Using PCMM Policy Servers*.

SAE Plug-Ins

Plug-ins are software programs that extend the capabilities of existing programs and make them more flexible. SRC plug-ins provide authentication, authorization, and tracking capabilities.

There are three types of plug-ins: internal, hosted, and external. Internal plug-ins communicate directly with the SAE. Hosted and external plug-ins implement a published Common Object Request Broker Architecture (CORBA)-based service provider interface (SPI), which means that anyone with access to the interface specification can create plug-ins that work with the SRC software. Figure 1 gives an overview of the plug-in architecture.

Figure 1: SAE Plug-In Architecture



Internal Plug-Ins

The SRC software provides internal plug-ins that perform a range of authentication, authorization, and tracking functions. With these plug-ins, you can, for example, authenticate subscribers, authorize subscriptions and sessions, authorize IP address requests from DHCP clients, track subscriber activity and service use, track quality of service (QoS) services and attach and remove QoS profiles as needed, and limit the number of authenticated subscribers who connect to an IP interface on the router.

Internal plug-ins implement an interface that communicates directly with the SAE. They have the following characteristics:

- Run within the SAE's Java Virtual Machine (JVM)
- Are started and stopped with the SAE
- Are implemented in Java

The core SRC software provides a set of internal plug-ins. See *SRC-PE Subscribers and Subscriptions Guide, Chapter 8, Overview of Plug-Ins Included with the SAE*.

External Plug-Ins

The SRC software includes the SAE CORBA plug-in SPI. This SPI allows you to implement external plug-ins in any language that supports CORBA (for example, Java, C++, Python), which makes it easy to integrate the SAE with operations support system (OSS) software written in a wide variety of languages and distributed across a variety of hardware and operating system platforms.

External plug-ins link a service provider's OSS with the SAE so that the OSS is notified of events in the life cycle of SAE sessions. For example, plug-ins can be notified when a subscriber attempts to log in and begins the authentication and authorization process. This notification makes it possible for the plug-in to consult general data and resource allocation information that is available to the OSS, and use that information to make authorization decisions.

The SPI also sends session-tracking events when sessions start, on an interim basis, and when sessions stop. Plug-ins can set session timeouts as a response to both session start and interim events. This capability enables the development of prepaid applications where the plug-in consults the subscriber's current account balance before it makes the decision to extend or reduce a session timeout.

External plug-ins have the following characteristics:

- Run outside the SAE's JVM, either in the same or in a different server
- Are implemented in any language that supports CORBA
- Communicate with the SAE using CORBA
- Support the admission control or prepaid demo plug-in, which can be purchased separately from the SRC software.

To configure the SAE for external plug-ins, see one of the following:

- *SRC-PE Subscribers and Subscriptions Guide, Chapter 9, Configuring Internal, External, and Synchronization Plug-Ins with the SRC CLI*
- *SRC-PE Subscribers and Subscriptions Guide, Chapter 10, Overview of Configuring Plug-Ins for Solaris Platforms*

The interface definition language (IDL) code and online documentation for the SAE CORBA Plug-In SPI is on the Juniper Networks Web site at
http://www.juniper.net/techpubs/software/management/sdx/api-index.html

The IDL code is also in the software distribution in the file *SDK/idl/sspPlugin.idl*. Online documentation for the *sspPlugin.idl* file is at */SDK/doc/idl/sspPlugin/html/index.html*.

Hosted Plug-Ins

Hosted plug-ins, like the external ones, implement the CORBA interface. Unlike the external ones, hosted plug-ins are instantiated (that is, hosted) by the SAE. As a result, they live in the same JVM process as the host SAE, which means that hosted plug-ins must be implemented in Java.

Hosted plug-ins have the following characteristics:

- Run within the SAE's JVM
- Communicate with SAE using CORBA
- Are started and stopped with SAE
- Are implemented using a published interface

Tracking and Controlling Subscriber and Service Sessions with SAE APIs

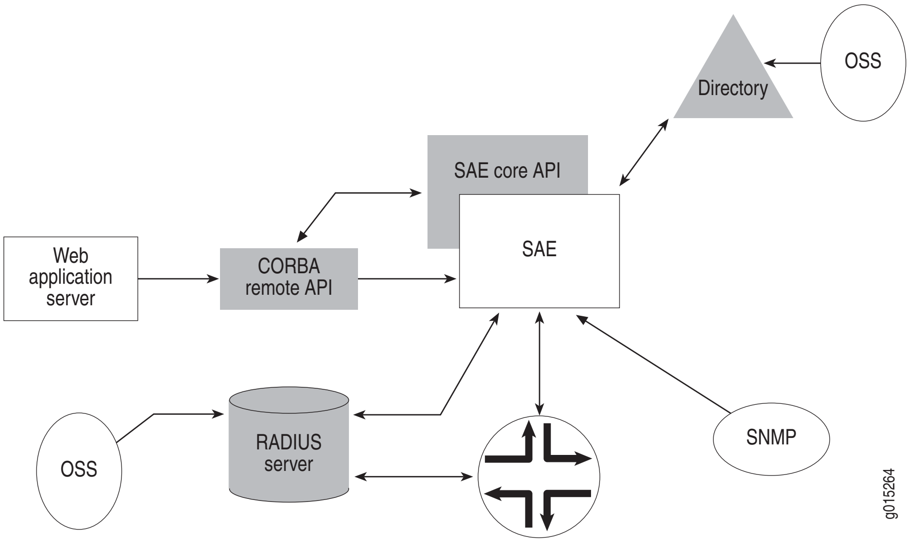
The SAE provides two public APIs:

- SAE core API
- SAE CORBA remote API

Through these interfaces, an external application can track and control subscriber and service sessions.

Figure 2 illustrates the SAE APIs.

Figure 2: SRC SAE APIs



SAE Core API

The SAE core API is used to control the behavior of the SRC software. There are many uses of the SAE core API. For example, it can be used to provide:

- Subscriber credentials (username/password)
- Requests for service activation/deactivation for a subscriber

This API can be used by a Java application running in the same JVM as the SAE. For example, you can access the SAE core API from plug-ins that are hosted by the SAE, or you can use the SAE core API to write your own extensions of the SAE remote interface by using CORBA or the SAE script interface modules.

SAE CORBA Remote API

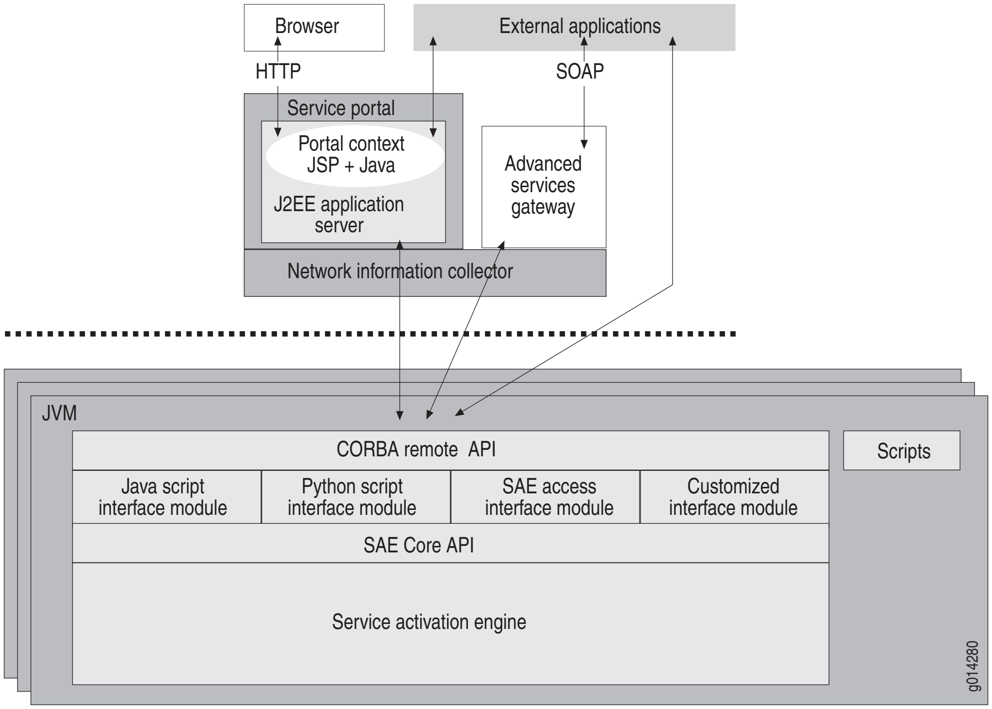
This API provides a way to use external applications with the SRC software (see Figure 3). All functions that are available through the SAE core API are available through the CORBA remote API. The remote API provides several remote interfaces that allow customization of the API for special needs. The remote interface comprises an interface module manager and a set of interface modules. We provide the following interface modules with the SRC software:

- SAE access interface module—Provides remote access to the SAE core API
- Java script interface module—Allows you to control the SAE with a Java script
- Python script interface module—Allows you to control the SAE with a Python script
- Event notification interface module—Allows you to integrate the SAE with external IP address managers

You can also create custom interface modules that allow external applications to extend the capabilities of the SAE. To do so, you must define the interface module in CORBA IDL and implement it in Java.

The remote interface publishes one object reference that acts as the interface module manager. External applications communicate through CORBA with the interface module manager to retrieve a particular interface module. That interface module runs in the same JVM as the SAE and has full access to the SAE core API.

Figure 3: Remote Interface on the SAE


Browser
External applications
HTTP
SOAP
Service portal
Portal context
JSP + Java
J2EE application server
Advanced services gateway
Network information collector
JVM
CORBA remote API
Scripts
Java script interface module
Python script interface module
SAE access interface module
Customized interface module
SAE Core API
Service activation engine
g014280

For more information about the SAE CORBA remote API, including the interfaces, properties, and methods, see the online documentation on the Juniper Networks Web site at
http://www.juniper.net/techpubs/software/management/sdx/api-index.html or in the SRC software distribution in *SDK/doc/idl/sae/html/index.html*.

SAE Accounting

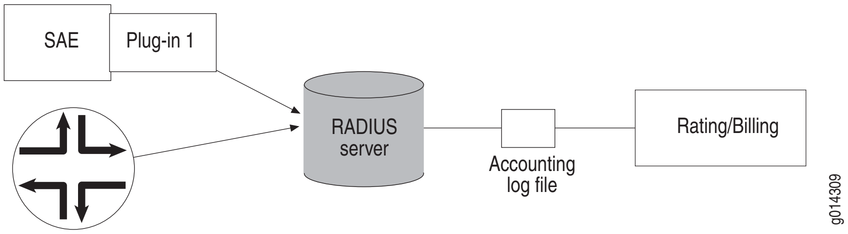
The router and the SAE generate RADIUS accounting records when subscribers access the Internet and use value-added services. The records are sent to RADIUS accounting servers and are logged in accounting log files, or they are sent to accounting flat files. External systems collect the accounting log files and feed them to a rating and billing system.

The SRC software allows a variety of accounting deployments. This section shows the standard deployment that we supply, a second option that does not depend on a RADIUS server, and a third option in which customers develop their own deployment by choosing a CORBA plug-in.

In the standard SRC deployment (see Figure 4), the router and the SAE are clients of the RADIUS accounting server. They pass subscriber accounting information to a designated RADIUS accounting server in an accounting request. The RADIUS accounting server receives the accounting request and creates accounting log files.

The SRC software works with other AAA RADIUS servers; however, we validate the SRC software only with Merit, Interlink RAD-Series AAA RADIUS Server, or Juniper Networks Steel-Belted Radius/SPE server.

Figure 4: Sending Accounting Data to a RADIUS Server



A second option, shown in Figure 5, uses an accounting flat file generated directly by the SAE, without a RADIUS server.

Figure 5: Sending Accounting Data to an Accounting File

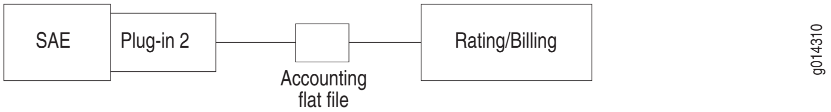
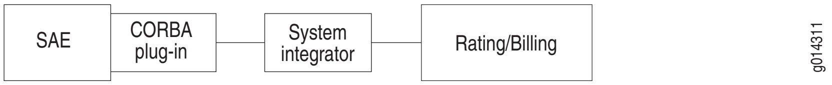


Figure 6 illustrates a third possibility, one in which the customer uses a CORBA plug-in of his or her own choice.

Figure 6: Customer Choice for SRC Accounting Deployment



Accounting Policy

The SAE defines the policies that control the network traffic for the subscriber based on the subscriber’s subscriptions. It also determines the accounting statistics collected for the subscribed service.

While defining the policies for a service, the SAE can choose the policy rules to be used for accounting per interface direction (ingress and egress). Statistics are collected for the chosen policy rules for the service and are sent to the RADIUS accounting server. The SAE can also decide not to collect any policy rule–specific statistics for the service. In this case, only session times are sent to the accounting system when the service is deactivated. When choosing multiple policy rules on traffic direction for statistics collection, the SAE summarizes the statistics by adding the individual values.

Subscription Process

After an outsourced service has been set up, subscribers can order primary access or value-added services from retailers, who in turn notify the wholesaler of the new end subscription. Conversely, accounting data is collected by the wholesaler and communicated to the retailer to provide enough data for the retailer to bill the subscriber.

The overall subscription process is simplified:

- The subscriber has no need to interact with another party or a device other than the router.
- When the subscriber goes to the Web portal and selects the service, the subscription activation is triggered.
- The subscriber’s portal page adjusts to display the new service.
- Accounting data is generated, identifying the service being tracked for the subscriber.

Tracking Subscriber Sessions

The intelligent service accounting function of the SRC software tracks the subscription activity for each subscriber and each service session. It collects usage information and passes the information to the appropriate rating and billing system.

Multiple service sessions can be activated simultaneously for a subscriber and can be tracked separately from an accounting standpoint.

Events are generated when service sessions are activated and deactivated, and during interim accounting updates.

Accounting Plug-Ins

Plug-ins allow service providers to easily extend the capabilities of their systems through the use of plug-in software. See *SAE Plug-Ins* on page 6.

Interim Accounting

The router and SAE generate interim accounting records for broadband primary services (through PPP) and value-added services, respectively. RADIUS servers log the interim records in their accounting log files when interim accounting is enabled.

The external rating system calculates the charges by using interim records instead of stop records for timeout sessions. The calculation occurs when the last record is interim and for open sessions whose last record at the end of a billing cycle is interim.

An accounting interim interval is defined for each service and applied to all subscriptions to that service. The router and SAE generate accounting requests with a status of interim for every period of time specified with the interim value.

The router receives an accounting interim value for a session through a RADIUS server when the router makes an authentication request. If the RADIUS server does not provide a value, then the router does not generate interim accounting records.

The SAE obtains an accounting interim value from the directory. When the accounting interim value is not stored, the SAE uses global values. When a value equals zero, the SAE does not generate interim accounting records.

Chapter 2

Configuring the SAE with the SRC CLI

This chapter describes how to use the SRC CLI to configure general SAE properties. You can use the SRC CLI to configure the SAE on a Solaris platform or on a C-series Controller.

To use the C-Web interface to configure an SAE on a Solaris platform or on a C-series Controller, see *Chapter 4, Managing SAE Data with the C-Web Interface*.

Topics in this chapter include:

- Configuring LDAP Access to Directory Data on page 13
- Storing Subscriber and Service Session Data on page 23
- Configuring the Session Store Feature on page 24
- Configuring the Number of Threads for Sessions on page 29

Configuring LDAP Access to Directory Data

The SRC software stores subscriber, service, persistent login, policy, router, and cached subscriber profiles and session data in a directory. The SAE uses LDAP to store and retrieve the data.

If you do not store data in the local directory, you need to configure the LDAP connections to the directories in which the data is stored. You can also select the filter that the SAE uses to search for subscriptions in the directory and directory eventing parameters for data stored in the directory.

The tasks to configure LDAP access to directory data are:

- (Optional) Configuring Access Through LDAPS to Service and Subscriber Data on page 14
- Configuring Access to Subscriber Data on page 15
- Configuring Access to Service Data on page 17
- Configuring Access to Policy Data on page 18
- Configuring Access to the Persistent Login Cache on page 19

- Configuring the Location of Network Device Data on page 21
- Enabling Automatic Discovery of Changes in SAE Configuration Data on page 21
- Setting the Timeout and Number of Events for SAE Directory Eventing on page 22

Configuring Access Through LDAPS to Service and Subscriber Data

You can secure connections between a router and an external directory that contains service data or subscriber data, and you can configure the router to use LDAPS when it connects to the same data source.

Use the following configuration statements to configure access through LDAPS to service data and subscriber data:

```
shared sae configuration ldap service-data {
    (ldaps);
}
```

```
shared sae configuration ldap subscriber-data {
    (ldaps);
}
```

To use LDAPS to secure connections between a router and an external directory:

1. Configure the directory connection from the SAE to use LDAPs. For example:

   ```
   user@host# set shared sae configuration ldap service-data ldaps
   ```

   ```
   user@host# set shared sae configuration ldap subscriber-data ldaps
   ```

2. In the router initialization script you specify the directory context.

   The */opt/UMC/sae/lib/poolPublisher.py* script and the */opt/UMC/sae/lib/IorPublisher.py* script provide examples of how to configure a directory context, For example, from the */opt/UMC/sae/lib/IorPublisher.py* script:

   ```
   dirContext = Ssp.registry.get('ServiceDataSource.component').getContext()
   ```

   In addition, you can change the directory context.

   For information about how to use InitialDirContext class or the DirContext class to specify directory context, see:

http://java.sun.com/j2se/1.4.2/docs/api/javax/naming/directory/InitialDirContext.html

http://java.sun.com/j2se/1.4.2/docs/api/javax/naming/directory/DirContext.html

Related Topics

- *Configuring Access to Subscriber Data* on page 15
- *Configuring Access to Service Data* on page 17

Configuring Access to Subscriber Data

Use the following configuration statements to configure access to subscriber data:

```
shared sae configuration ldap subscriber-data {
    subscription-loading-filter (subscriberRefFilter | objectClassFilter);
    load-subscriber-schedules;
    login-cache-dn login-cache-dn;
    session-cache-dn session-cache-dn;
    server-address server-address;
    dn dn;
    authentication-dn authentication-dn;
    password password;
    directory-eventing;
    polling-interval polling-interval;
    (ldaps);
}
```

To configure SAE access to subscriber data:

1. From configuration mode, access the configuration statement that configures SAE access to subscriber data in the directory. In this sample procedure, the subscriber data is configured in the se-region group.

   ```
   user@host# edit shared sae group se-region configuration ldap subscriber-data
   ```

2. Select the filter that the SAE uses to search for subscriptions in the directory when the SAE loads a subscription to a subscriber reference filter.

   ```
   [edit shared sae group se-region configuration ldap subscriber-data]
   user@host# set subscription-loading-filter (subscriberRefFilter | objectClassFilter)
   ```

3. (Optional) Enable loading of subscriber schedules.

   ```
   [edit shared sae group se-region configuration ldap subscriber-data]
   user@host# set load-subscriber-schedules
   ```

4. Specify the subtree in the directory in which subscriber information is stored.

   ```
   [edit shared sae group se-region configuration ldap subscriber-data]
   user@host# set login-cache-dn login-cache-dn
   ```

5. Specify the subtree in the directory in which persistent session data is cached.

   ```
   [edit shared sae group se-region configuration ldap subscriber-data]
   user@host# set session-cache-dn session-cache-dn
   ```

6. (Optional) Specify the directory server that stores subscriber information.

   ```
   [edit shared sae group se-region configuration ldap subscriber-data]
   user@host# set server-address server-address
   ```

7. Specify the subtree in the directory where subscriber data is cached.

   ```
   [edit shared sae group se-region configuration ldap subscriber-data]
   user@host# set dn dn
   ```

8. (Optional) Specify the DN that the SAE uses to authenticate access to the directory server.

   ```
   [edit shared sae group se-region configuration ldap subscriber-data]
   user@host# set authentication-dn authentication-dn
   ```

9. (Optional) Specify the password used to authenticate access to the directory server.

   ```
   [edit shared sae group se-region configuration ldap subscriber-data]
   user@host# set password password
   ```

10. (Optional) Enable automatic discovery of changes in subscriber profiles.

    ```
    [edit shared sae group se-region configuration ldap subscriber-data]
    user@host# set directory-eventing
    ```

11. Set the frequency for checking the directory for updates.

    ```
    [edit shared sae group se-region configuration ldap subscriber-data]
    user@host# set polling-interval polling-interval
    ```

12. Enable LDAPS as the secure protocol for connections to the server that stores subscriber data.

    ```
    [edit shared sae group se-region configuration ldap subscriber-data]
    user@host# set ldaps
    ```

13. (Optional) Verify your configuration.

    ```
    [edit shared sae group se-region configuration ldap subscriber-data]
    user@host# show
    subscription-loading-filter objectClassFilter;
    load-subscriber-schedules;
    login-cache-dn o=users,<base>;
    session-cache-dn o=PersistentSessions,<base>;
    server-address 127.0.0.1;
    dn o=users,<base>;
    authentication-dn cn=ssp,o=components,o=operators,<base>;
    password ********;
    directory-eventing;
    polling-interval 30;
    ldaps;
    ```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*
- *Configuring Access Through LDAPS to Service and Subscriber Data* on page 14

Configuring Access to Service Data

Use the following configuration statements to configure access to service data:

```
shared sae configuration ldap service-data {
    server-address server-address;
    dn dn;
    authentication-dn authentication-dn;
    password password;
    directory-eventing;
    polling-interval polling-interval;
    (ldaps);
}
```

To configure SAE access to service data:

1. From configuration mode, access the configuration statement that configures SAE access to service data in the directory. In this sample procedure, the service data is configured in the se-region group.

   ```
   user@host# edit shared sae group se-region configuration ldap service-data
   ```

2. (Optional) Specify the directory server that stores service data.

   ```
   [edit shared sae group se-region configuration ldap service-data]
   user@host# set server-address server-address
   ```

3. Specify the subtree in the directory where service data is cached.

   ```
   [edit shared sae group se-region configuration ldap service-data]
   user@host# set dn dn
   ```

4. (Optional) Specify the DN that the SAE uses to authenticate access to the directory server.

   ```
   [edit shared sae group se-region configuration ldap service-data]
   user@host# set authentication-dn authentication-dn
   ```

5. (Optional) Specify the password used to authenticate access to the directory server.

   ```
   [edit shared sae group se-region configuration ldap service-data]
   user@host# set password password
   ```

6. (Optional) Enable or disable automatic discovery of changes to service data.

   ```
   [edit shared sae group se-region configuration ldap service-data]
   user@host# set directory-eventing
   ```

7. Set the frequency for checking the directory for updates.

   ```
   [edit shared sae group se-region configuration ldap service-data]
   user@host# set polling-interval polling-interval
   ```

8. Enable LDAPS as the secure protocol for connections to the server that stores service data.

   ```
   edit shared sae group se-region configuration ldap service-data]
   user@host# set ldaps
   ```

9. (Optional) Verify your configuration.

   ```
   [edit shared sae group se-region configuration ldap service-data]
   user@host# show
   server-address 10.10.45.3;
   dn <base>;
   authentication-dn <base>;
   password ********;
   directory-eventing;
   polling-interval 30;
   ldaps;
   ```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*
- *Configuring Access Through LDAPS to Service and Subscriber Data* on page 14

Configuring Access to Policy Data

Use the following configuration statements to configure access to policy data:

```
shared sae configuration ldap policy-data {
    policy-dn policy-dn;
    parameter-dn parameter-dn;
    directory-eventing;
    polling-interval polling-interval;
}
```

To configure SAE access to subscriber data:

1. From configuration mode, access the configuration statement that configures SAE access to policy data in the directory. In this sample procedure, the policy data is configured in the se-region group.

   ```
   user@host# edit shared sae group se-region configuration ldap policy-data
   ```

2. Specify the subtree in the directory in which policy data stored.

   ```
   [edit shared sae group se-region configuration ldap policy-data]
   user@host# set policy-dn policy-dn
   ```

3. Specify the subtree in the directory in which policy parameter data is cached.

   ```
   [edit shared sae group se-region configuration ldap policy-data]
   user@host# set parameter-dn parameter-dn
   ```

4. (Optional) Enable or disable automatic discovery of changes to policy data.

   ```
   [edit shared sae group se-region configuration ldap policy-data]
   user@host# set directory-eventing
   ```

5. Set the frequency for checking the directory for updates.

   ```
   [edit shared sae group se-region configuration ldap policy-data]
   user@host# set polling-interval polling-interval
   ```

6. (Optional) Verify your configuration.

   ```
   [edit shared sae group se-region configuration ldap policy-data]
   user@host# show
   policy-dn o=Policy,<base>;
   parameter-dn o-Parameters,<base>;
   directory-eventing;
   polling-interval 30;
   ```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*

Configuring Access to the Persistent Login Cache

Use the following configuration statements to configure access to persistent login cache data:

```
shared sae configuration ldap persistent-login-cache {
    server-address server-address;
    dn dn;
    authentication-dn authentication-dn;
    password password;
    directory-eventing;
    polling-interval polling-interval;
    (ldaps);
}
```

To configure SAE access to persistent login cache data:

1. From configuration mode, access the configuration statement that configures SAE access to persistent login cache data in the directory. In this sample procedure, the persistent login cache data is configured in the se-region group.

   ```
   user@host# edit shared sae group se-region configuration ldap
   persistent-login-cache
   ```

2. (Optional) Specify the directory server that stores service data.

   ```
   [edit shared sae group se-region configuration ldap persistent-login-cache]
   user@host# set server-address server-address
   ```

3. Specify the subtree in the directory where persistent login cache data is cached.

   ```
   [edit shared sae group se-region configuration ldap persistent-login-cache]
   user@host# set dn dn
   ```

4. (Optional) Specify the DN that the SAE uses to authenticate access to the directory server.

   ```
   [edit shared sae group se-region configuration ldap persistent-login-cache]
   user@host# set authentication-dn authentication-dn
   ```

5. (Optional) Specify the password used to authenticate access to the directory server.

   ```
   [edit shared sae group se-region configuration ldap persistent-login-cache]
   user@host# set password password
   ```

6. (Optional) Enable automatic discovery of changes to persistent login cache data.

   ```
   [edit shared sae group se-region configuration ldap persistent-login-cache]
   user@host# set directory-eventing
   ```

7. Set the frequency for checking the directory for updates.

   ```
   [edit shared sae group se-region configuration ldap persistent-login-cache]
   user@host# set polling-interval polling-interval
   ```

8. Enable LDAPS as the secure protocol for connections to the server that stores persistent login cache data.

   ```
   [edit shared sae group se-region configuration ldap persistent-login-cache]
   user@host# set ldaps
   ```

9. (Optional) Verify your configuration.

   ```
   [edit shared sae group se-region configuration ldap persistent-login-cache]
   user@host# show
   dn "o=authCache, <base>";
   directory-eventing;
   polling-interval 30;
   ldaps;
   ```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*

Configuring the Location of Network Device Data

Use the following configuration statement to configure access to network device data:

```
shared sae configuration ldap {
    network-dn network-dn;
}
```

To configure SAE access to network device data:

1. From configuration mode, access the configuration statement that configures SAE access to network device data in the directory. In this sample procedure, the network device data is configured in the se-region group.

   ```
   user@host# edit shared sae group se-region configuration ldap
   ```

2. Specify the subtree in the directory where network device data is stored.

   ```
   [edit shared sae group se-region configuration ldap]
   user@host# set network-dn network-dn
   ```

3. Verify your configuration.

   ```
   [edit shared sae group se-region configuration ldap]
   user@host# show network-dn
   network-dn o=Network,<base>;
   ```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*

Enabling Automatic Discovery of Changes in SAE Configuration Data

Use the following configuration statement to enable automatic discovery of changes in SAE configuration data:

```
shared sae configuration ldap {
    enable-directory-eventing;
}
```

To enable automatic discovery of changes in SAE configuration data:

1. From configuration mode, access the configuration statement that enables automatic discovery of changes in SAE configuration data in the directory. In this sample procedure, automatic discovery is configured in the se-region group.

   ```
   user@host# edit shared sae group se-region configuration ldap
   ```

2. Enable automatic discovery of changes to SAE configuration data.

   ```
   [edit shared sae group se-region configuration ldap]
   user@host# enable-directory-eventing
   ```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*

Setting the Timeout and Number of Events for SAE Directory Eventing

Use the following configuration statements to set the directory eventing timeout and the number of simultaneous events that the SAE can receive from the directory:

```
shared sae configuration ldap directory-eventing {
    timeout timeout;
    dispatcher-pool-size dispatcher-pool-size;
}
```

To configure the directory eventing timeout and the number of simultaneous events that the SAE can receive from the directory:

1. From configuration mode, access the configuration statement that configures SAE directory eventing. In this sample procedure, directory eventing is configured in the se-region group.

   ```
   user@host# edit shared sae group se-region configuration ldap
   directory-eventing
   ```

2. Specify the maximum time that the directory eventing system waits for the directory to respond.

   ```
   [edit shared sae group se-region configuration ldap directory-eventing]
   user@host# set timeout timeout
   ```

3. Specify the number of events that the SAE can receive from the directory simultaneously.

   ```
   [edit shared sae group se-region configuration ldap directory-eventing]
   user@host# set dispatcher-pool-size dispatcher-pool-size
   ```

4. (Optional) Verify your configuration.

   ```
   [edit shared sae group se-region configuration ldap directory-eventing]
   user@host# show
   timeout 60;
   dispatcher-pool-size 1000;
   ```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*

Storing Subscriber and Service Session Data

To aid in recovering from an SAE failover, the SAE stores subscriber and service session data in flat files on the SAE host. 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. After the data has been written to disk, it can survive a server reboot.

You can configure how the SAE stores session data for JUNOSe routers, JUNOS routing platforms, simulated routers, and *PacketCable Multimedia Specification* (PCMM) devices.

Session Store Files

Session store files are numbered flat files. Session store files are located in a directory on the SAE host. You can configure the size of session store files. After the maximum size has been reached, the session store creates a new file and begins writing data to the new file.

Store operations, such as adding a session to the store (put store operations) or removing a session from the store (remove store operations), are queued in a buffer before they are written to the session store file. You can configure parameters that determine when the session store writes a queue to a session store file.

Session store files are deleted if they have not been modified and if no session activity has taken place for one week. All the data files that contain the sessions associated with a particular virtual router are deleted at the same time.

Active and Passive Session Stores

You can have a community of SAEs and duplicate session store data on each SAE in the community in case of an SAE failover. SAE communities are made up of SAEs that you configure as connected SAEs for a virtual router object.

SAEs in a community are given the role of either active SAE or passive SAE. The active SAE keeps session data up to date within the community. Each active session store opens a Transmission Control Protocol (TCP) connection to its passive SAE. The TCP connection triggers the creation of a passive session store in that SAE. When the active session store writes operations to the session store file, it passes them to passive session stores on all SAEs in the community.

When you modify a community, wait for passive session stores on the new community members to be updated before you shut down the currently 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.

Standby SAEs

In a community of SAEs, one SAE can provide redundancy for the active SAE. The redundant (standby) SAE connects to the active SAE through a COPS-PR connection. State as well as session data is replicated from the active SAE to the standby SAE to reduce the failover time from one SAE to another.

A standby SAE can respond to SAE failures and connection failures between an SAE and a JUNOSe router. Connection failures between an active SAE and a standby SAE may not be immediately detected, because each SAE continues to function for a period of time. When a standby SAE does detect that state information is different on the two SAEs, it resynchronizes data between the two.

NOTE: We recommend that you use a highly reliable and available connection between an active SAE and a standby SAE to ensure availability of the two SAEs.

Session Store File Rotation

The session store periodically rotates the session store files. During rotation, the session store copies put store operations for live sessions from the oldest file to the end of the newest file. (Live sessions are sessions that have been created but not yet deleted.) It then deletes the oldest file. Sessions are rotated in batches, and you can configure the number of sessions that are rotated at the same time, and how much disk space is used by live sessions before files are rotated. No session store activity can take place while a batch of sessions is rotated.

Configuring the Session Store Feature

You can configure three things for the session store feature:

- Configure session store parameters for a router or device driver. See *Configuring Session Store Parameters for a Device Driver* on page 24.
- Configure global session store parameters that are shared by all session store instances (active or passive) on the SAE. See *Configuring Global Session Store Parameters* on page 27.
- Reduce the size of session objects that the SAE sends across the network for the session store feature. See *Reducing the Size of Objects for the Session Store Feature* on page 28.

Configuring Session Store Parameters for a Device Driver

Use the following configuration statements to configure session store parameters within a device driver configuration:

```
shared sae configuration driver ( junos | junose | pcmm | simulated | third-party )
session-store {
    maximum-queue-age maximum-queue-age;
    maximum-queued-operations maximum-queued-operations;
    maximum-queue-size maximum-queue-size;
    maximum-file-size maximum-file-size;
    minimum-disk-space-usage minimum-disk-space-usage;
    rotation-batch-size rotation-batch-size;
    maximum-session-size maximum-session-size;
    disk-load-buffer-size disk-load-buffer-size;
    network-buffer-size network-buffer-size;
    retry-interval retry-interval;
    communications-timeout communications-timeout;
    load-timeout load-timeout;
    idle-timeout idle-timeout;
    maximum-backlog-ratio maximum-backlog-ratio;
    minimum-backlog minimum-backlog;
}
```

To configure session store parameters within a device driver configuration:

1. From configuration mode, access the configuration statement that configures the session store for your device driver. In this sample procedure, the session store for a JUNOS device driver is configured in the se-region group.

   ```
   user@host# edit shared sae group se-region configuration driver junos session-store
   ```

2. (Optional) Specify the maximum age that a queue of buffered store operations (such as adding a session to the store or removing a session from the store) can reach before the queue is written to a session store file.

   ```
   [edit shared sae group se-region configuration driver junos session-store]
   user@host# set maximum-queue-age maximum-queue-age
   ```

3. (Optional) Specify the number of buffered store operations that are queued before the queue is written to a session store file.

   ```
   [edit shared sae group se-region configuration driver junos session-store]
   user@host# set maximum-queued-operations maximum-queued-operations
   ```

4. (Optional) Specify the maximum size that a queue of buffered store operations can reach before the queue is written to a session store file.

   ```
   [edit shared sae group se-region configuration driver junos session-store]
   user@host# set maximum-queue-size maximum-queue-size
   ```

5. (Optional) Specify the maximum size of session store files.

   ```
   [edit shared sae group se-region configuration driver junos session-store]
   user@host# set maximum-file-size maximum-file-size
   ```

6. (Optional) Specify the percentage of space in all session store files that is used by live sessions.

   ```
   [edit shared sae group se-region configuration driver junos session-store]
   user@host# set minimum-disk-space-usage minimum-disk-space-usage
   ```

7. (Optional) Specify the number of sessions that are rotated from the oldest file to the newest file at the same time that the oldest session store file is rotated.

   ```
   [edit shared sae group se-region configuration driver junos session-store]
   user@host# set rotation-batch-size rotation-batch-size
   ```

8. (Optional) Specify the maximum size of a single subscriber or service session.

    ```
    [edit shared sae group se-region configuration driver junos session-store]
    user@host# set maximum-session-size maximum-session-size
    ```

9. (Optional) Specify the size of the buffer that is used to load all of a session store's files from disk at startup.

    ```
    [edit shared sae group se-region configuration driver junos session-store]
    user@host# set disk-load-buffer-size disk-load-buffer-size
    ```

10. (Optional) Specify the size of the buffer that holds messages or message segments that are waiting to be sent to passive session stores.

    ```
    [edit shared sae group se-region configuration driver junos session-store]
    user@host# set network-buffer-size network-buffer-size
    ```

11. (Optional) Specify the time interval between attempts by the active session store to connect to missing passive session stores.

    ```
    [edit shared sae group se-region configuration driver junos session-store]
    user@host# set retry-interval retry-interval
    ```

12. (Optional) Specify the amount of time that a session store waits before closing when it is blocked from reading or writing a message.

    ```
    [edit shared sae group se-region configuration driver junos session-store]
    user@host# set communications-timeout communications-timeout
    ```

13. (Optional) Specify the time that an active session store waits for a passive session store or a passive session store waits for an active session store to load its data from disk before it closes the connection to the session store.

    ```
    [edit shared sae group se-region configuration driver junos session-store]
    user@host# set load-timeout load-timeout
    ```

14. (Optional) Specify the time that a passive session store waits for activity from the active session store before it closes the connection to the active session store.

    ```
    [edit shared sae group se-region configuration driver junos session-store]
    user@host# set idle-timeout idle-timeout
    ```

15. (Optional) Specify when the active session store closes the connection to a passive session store because of a backlog of messages waiting to be sent.

    ```
    [edit shared sae group se-region configuration driver junos session-store]
    user@host# set maximum-backlog-ratio maximum-backlog-ratio
    ```

    ```
    [edit shared sae group se-region configuration driver junos session-store]
    user@host# set minimum-backlog minimum-backlog
    ```

16. (Optional) Verify your configuration.

```
[edit shared sae group se-region configuration driver junos session-store]
user@host# show
maximum-queue-age 5000;
maximum-queued-operations 50;
maximum-queue-size 51050;
maximum-file-size 25000000;
minimum-disk-space-usage 25;
rotation-batch-size 50;
maximum-session-size 10000;
disk-load-buffer-size 1000000;
network-buffer-size 51050;
retry-interval 5000;
communications-timeout 60000;
load-timeout 420000;
idle-timeout 3600000;
maximum-backlog-ratio 1.5;
minimum-backlog 5000000;
```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*

Configuring Global Session Store Parameters

This section describes how to configure global session store parameters that are shared by all session store instances (active or passive) on the SAE. You can also configure session store parameters within a device driver configuration. See *Configuring Session Store Parameters for a Device Driver* on page 24.

Use the following configuration statements to configure global session store parameters.

```
shared sae configuration driver session-store {
    ip-address ip-address;
    port port;
    root-directory root-directory;
}
```

To configure global session store parameters:

1. From configuration mode, access the configuration statement that configures the global session store parameters. In this sample procedure, the global session store is configured in the se-region group.

   ```
   user@host# edit shared sae group se-region configuration driver session-store
   ```

2. (Optional) Specify the IP address or hostname that the session store infrastructure on this SAE uses to listen for incoming TCP connections from active session stores.

   ```
   [edit shared sae group se-region configuration driver session-store]
   user@host# set ip-address ip-address
   ```

3. (Optional) Specify the TCP port number on which the session store infrastructure on this SAE listens for incoming connections from active session stores.

   [edit shared sae group se-region configuration driver session-store]
   user@host# **set port** *port*

4. (Optional) Specify the root directory in which the session store creates files.

   [edit shared sae group se-region configuration driver session-store]
   user@host# **set root-directory** *root-directory*

5. (Optional) Verify your configuration.

```
[edit shared sae group se-region configuration driver session-store]
user@host# show
ip-address 10.10.70.0;
port 8820;
root-directory var/sessionStore;
```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*

Reducing the Size of Objects for the Session Store Feature

You can use serialized data compression to reduce the size of sessions objects that the SAE sends across the network for the session store feature. Enabling this property reduces the size of objects, but increases the CPU load on the SAE.

Use the following configuration statement to specify whether or not session objects are compressed.

```
shared sae configuration {
    compress-session-data;
}
```

To specify whether or not session objects are compressed:

1. From configuration mode, access the sae configuration. In this sample procedure, data compression is configured in the se-region group.

   user@host# **edit shared sae group se-region configuration**

2. Enable reducing the size of session objects (subscriber and service sessions) that the SAE sends across the network for the session store feature.

   [edit shared sae group se-region configuration]
   user@host# **set compress-session-data**

3. (Optional) Verify your configuration.

```
[edit shared sae group se-region configuration]
user@host# show compress-session-data
compress-session-data;
```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*

Configuring the Number of Threads for Sessions

Use the following configuration statement to set the number of threads used for session-related activity.

```
shared sae configuration session-job-manager {
    number-of-threads number-of-threads;
}
```

To configure the number of threads used to handle session-related activity:

1. From configuration mode, access the session job manager configuration. In this sample procedure, the number of threads is configured in the se-region group.

   ```
   user@host# edit shared sae group se-region configuration session-job-manager
   ```

2. Specify the number of threads used for session-related activity.

   ```
   [edit shared sae group se-region configuration session-job-manager]
   user@host# set number-of-threads number-of-threads
   ```

3. (Optional) Verify your configuration.

   ```
   [edit shared sae group se-region configuration session-job-manager]
   user@host# show
   number-of-threads 10;
   ```

Chapter 3

Managing SAE Data with the SRC CLI

This chapter describes how to use the CLI to manage the SAE on a Solaris platform or on the C-series Controller.

You can also use the C-Web interface to manage the SAE. See *Managing SAE Data with the C-Web Interface* on page 37.

Topics include:

Commands to Manage SAE

You can use the following operational mode commands to manage SAE data:

- clear sae directory-blacklist
- clear sae registered equipment
- clear sae registered login
- request sae java-garbage-collection
- request sae load configuration
- request sae load domain-map
- request sae load interface-classification

- request sae load services
- request sae load subscriptions
- request sae modify device failover
- request sae shutdown device
- show sae directory-blacklist
- show sae drivers
- show sae registered equipment
- show sae registered login

For detailed information about each command, see the *SRC CLI Command Reference*.

Reloading the SAE Data

You can reload specified configuration components. You can reload the SAE server's current configuration for:

- SAE configuration
- Services
- Subscriptions
- Interface classifiers
    - Domain map

To view configuration information, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 15, Monitoring SAE Data with the SRC CLI*.

Reloading the SAE Configuration

To reload the SAE configuration data from the directory:

```
user@host> request sae load configuration
```

The new configuration takes effect immediately.

Reloading Services

To reload the services, scopes, virtual routers, policies, service mutex groups, and service schedules from the directory:

```
user@host> request sae load services
```

Related service sessions are activated, deactivated, or reactivated as needed.

Reloading Subscriptions

To reload all subscriptions from the directory:

```
user@host> request sae load subscriptions
```

Related service sessions are activated, deactivated, or reactivated as needed.

Reloading Interface Classification Scripts

To reload the interface classification scripts from the directory, and apply the result of the interface classification changes to the router:

```
user@host> request sae load interface-classification
```

Reloading Domain Maps

To reload the mapping of domain names to retailer entries:

```
user@host> request sae load domain-map
```

This mapping is made available to the SAE's subscriber classification script.

Updating Memory Usage

To ensure that changes are updated, run Java Virtual Machine (JVM) garbage collection. This process frees memory and results in more accurate Heap in Use statistics.

```
user@host> request sae java-garbage-collection
```

Removing the Directory Blacklist

To remove the directory blacklist:

1. Issue the `show sae directory-blacklist` command to view information about the directory blacklist.
2. Issue the `clear sae directory-blacklist command` to remove the directory blacklist.

Removing Login Registrations

You can delete all login registrations, or you can delete a specific registration. For information about login registrations, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 4, Configuring Subscriber–Related Properties on the SAE with the SRC CLI*.

To remove login registrations:

1. Issue the show sae registered login command to view the login registrations.
2. Issue the clear sae registered login command to remove all login registrations.
   - To remove a specific registration, use the mac-address option and specify the media access control (MAC) address for the registration.

     ```
     user@host> clear sae registered login mac-address mac-address
     ```

   - To specify that no confirmation is requested before the software deletes the registration entries, use the force option.

     ```
     user@host> clear sae registered login force
     user@host> clear sae registered login mac-address mac-address force
     ```

Removing Equipment Registrations

You can delete all equipment registrations, or you can delete a specific registration. The demonstration residential portal included with the SRC Application Library provides an example of how to use equipment registration.

To remove equipment registrations:

1. Issue the show sae registered equipment command to view the equipment registrations.
2. Issue the clear sae registered equipment command to remove all equipment registrations.
   - To remove a specific registration, use the mac-address option and specify the media access control (MAC) address for the registration.

     ```
     user@host> clear sae registered equipment mac-address mac-address
     ```

   - To specify that no confirmation is requested before the software deletes the registration entries, use the force option.

     ```
     user@host> clear sae registered equipment force
     user@host> clear sae registered equipment mac-address mac-address force
     ```

Modifying Failover Server Parameters

To modify failover server parameters:

1. Issue the show sae drivers brief command to view the router or device instances.

2. Issue the request sae modify device failover virtual-router-name *virtual-router-name* command to modify failover server parameters.

    - (Optional) To modify the IP address of an alternate SAE server to which a router can reconnect when this driver closes its connection, use the ip-address option. This option is not applicable to the PCMM device driver.

      ```
      user@host> request sae modify device failover virtual-router-name
      virtual-router-name ip-address ip-address
      ```

    - (Optional) To modify the port of an alternate SAE server to which a router can reconnect when this driver closes its connection, use the tcp-port option. This option is not applicable to the PCMM device driver.

      ```
      user@host> request sae modify device failover virtual-router-name
      virtual-router-name tcp-port tcp-port
      ```

    - (Optional) To specify whether the device driver sends its own failover IP address and port to the router when it closes its connection, use the use-failover-server option. This option is not applicable to the PCMM device driver.

      ```
      user@host> request sae modify device failover virtual-router-name
      virtual-router-name use-failover-server
      ```

    - (Optional) To specify that no confirmation is requested before the software modifies the parameters, use the force option.

      ```
      user@host> request sae modify device failover virtual-router-name
      virtual-router-name force
      ```

      ```
      user@host> request sae modify device failover virtual-router-name
      virtual-router-name ip-address ip-address force
      ```

      ```
      user@host> request sae modify device failover virtual-router-name
      virtual-router-name tcp-port tcp-port force
      ```

      ```
      user@host> request sae modify device failover virtual-router-name
      virtual-router-name use-failover-server force
      ```

Shutting Down the Device Drivers

To shut down the specified router or device instance:

1. Issue the show sae drivers brief command to view the router or device instances.
2. Issue the request sae shutdown device command to shut down all device drivers.
   - To shut down specific drivers managing a virtual router, use the filter option and specify all or part of the name of the virtual router.

     ```
     user@host> request sae shutdown device filter filter
     ```

   - To specify that no confirmation is requested before the software shuts down the device drivers, use the force option.

     ```
     user@host> request sae shutdown device force
     user@host> request sae shutdown device filter filter force
     ```

Chapter 4

Managing SAE Data with the C-Web Interface

This chapter describes how to use the C-Web interface to manage the SAE on a Solaris platform or on the C-series Controller.

You can also use the SRC CLI to manage the SAE. See *Managing SAE Data with the SRC CLI* on page 31.

Topics in this chapter include:

Reloading the SAE Data with the C-Web Interface

You can reload specified configuration components. You can reload the SAE server's current configuration for:

- SAE configuration
- Services
- Subscriptions
- Interface classifiers
- Domain map

To view configuration information, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 15, Monitoring SAE Data with the SRC CLI*.

Reloading the SAE Configuration

To reload the SAE configuration data from the directory:

1. Click **Manage > Request > SAE > Load > Configuration**.

   The Configuration pane appears.

2. Enter information as described in the Help text in the main pane, and click **OK**.

The new configuration takes effect immediately.

Reloading Services

To reload the services, scopes, virtual routers, policies, service mutex groups, and service schedules from the directory:

1. Click **Manage > Request > SAE > Load > Services**.

   The Services pane appears.

2. Enter information as described in the Help text in the main pane, and click **OK**.

Related service sessions are activated, deactivated, or reactivated as needed.

Reloading Subscriptions

To reload all subscriptions from the directory:

1. Click **Manage > Request > SAE > Load > Subscriptions**.

   The Subscriptions pane appears.

2. Enter information as described in the Help text in the main pane, and click **OK**.

Related service sessions are activated, deactivated, or reactivated as needed.

Reloading Interface Classification Scripts

To reload the interface classification scripts from the directory, and apply the result of the interface classification changes to the router:

1. Click **Manage > Request > SAE > Load > Interface Classification**.

   The Interface Classification pane appears.

2. Enter information as described in the Help text in the main pane, and click **OK**.

Reloading Domain Maps

To reload the mapping of domain names to retailer entries:

1. Click **Manage > Request > SAE > Load > Domain Map**.

   The Domain Map pane appears.

2. Enter information as described in the Help text in the main pane, and click **OK**.

This mapping is made available to the SAE's subscriber classification script.

Updating Memory Usage with the C-Web Interface

To ensure that changes are updated, run Java Virtual Machine (JVM) garbage collection. This process frees memory and results in more accurate Heap in Use statistics.

To update memory usage:

1. Click **Manage > Request > SAE > Java Garbage Collection**.

   The Java Garbage Collection pane appears.

2. Enter information as described in the Help text in the main pane, and click **OK**.

Removing the Directory Blacklist with the C-Web Interface

To remove the directory blacklist:

1. To view information about the directory blacklist:
   a. Click **Monitor > SAE > Directory Blacklist**.

      The Directory Blacklist pane appears.

   b. Enter information as described in the Help text in the main pane, and click **OK**.
2. To remove the directory blacklist:
   a. Click **Manage > Clear > SAE > Directory Blacklist**.

      The Directory Blacklist pane appears.

   b. Enter information as described in the Help text in the main pane, and click **OK**.

Removing Login Registrations with the C-Web Interface

You can delete all login registrations, or you can delete a specific registration. For information about login registrations, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 4, Configuring Subscriber–Related Properties on the SAE with the SRC CLI*.

To remove login registrations:

1. To view the login registrations:
   a. Click **Monitor > SAE > Registered > Login**.

      The Login pane appears.

   b. Enter information as described in the Help text in the main pane, and click **OK**.
2. To remove login registrations:
   a. Click **Manage > Clear > SAE > Registered > Login**.

      The Login pane appears.

   b. Enter information as described in the Help text in the main pane, and click **OK**.

Removing Equipment Registrations with the C-Web Interface

You can delete all equipment registrations, or you can delete a specific registration. The demonstration residential portal included with the SRC Application Library provides an example of how to use equipment registration.

To remove equipment registrations:

1. To view the login registrations:
   a. Click **Monitor > SAE > Registered > Equipment**.

      The Equipment pane appears.

   b. Enter information as described in the Help text in the main pane, and click **OK**.
2. To remove login registrations:
   a. Click **Manage > Clear > SAE > Registered > Equipment**.

      The Equipment pane appears.

   b. Enter information as described in the Help text in the main pane, and click **OK**.

Modifying Failover Server Parameters with the C-Web Interface

To modify failover server parameters:

1. To view the router or device instances:
   a. Click **Monitor > SAE > Drivers**.

      The Drivers pane appears.
   b. Enter information as described in the Help text in the main pane, and click **OK**.
2. To modify failover server parameters:
   a. Click **Manage > SAE > Request > Modify > Device > Failover**.

      The Failover pane appears.
   b. Enter information as described in the Help text in the main pane, and click **OK**.

Shutting Down the Device Drivers with the C-Web Interface

To shut down the specified router or device instance:

1. To view the router or device instances:
   a. Click **Monitor > SAE > Drivers**.

      The Drivers pane appears.
   b. Enter information as described in the Help text in the main pane, and click **OK**.
2. To shut down all device drivers:
   a. Click **Manage > SAE > Request > Shutdown > Device**.

      The Device pane appears.
   b. Enter information as described in the Help text in the main pane, and click **OK**.

Part 2

Using Juniper Networks Routers in the SRC Network

Chapter 5

Using JUNOSe Routers in the SRC Network with the SRC CLI

This chapter describes how to use the SRC CLI to set up the SRC software and how to set up a JUNOSe router so that the router can be used in the SRC network. It also shows how to monitor the interactions between the SAE and the JUNOSe router and how to troubleshoot SRC problems on the router.

You can also use the following to configure JUNOSe routers:

- To use the C-Web interface, see *SRC-PE C-Web Interface Configuration Guide, Chapter 17, Using JUNOSe Routers in the SRC Network with the C-Web Interface*.
- To use the Solaris platform, see *Chapter 6, Using JUNOSe Routers in the SRC Network with a Solaris Platform*.

Topics in this chapter include:

COPS Connection Between JUNOSe Routers and the SAE

Configuring the SRC client on a JUNOSe router opens a Common Open Policy Service (COPS) protocol layer connection to the SAE. When the SRC client software establishes a TCP/IP connection to the SAE, the SAE starts to manage the JUNOSe router. Subsequently, the SRC client sends configuration changes made on the JUNOSe router to the SAE, and the SAE updates SRC configurations for services and policies accordingly.

The SAE supports two versions of COPS:

- COPS usage for policy provisioning (COPS-PR)
- COPS External Data Representation Standard (COPS-XDR)

The version of COPS that you use depends on the version of COPS that your JUNOSe router supports. When you set up your JUNOSe router to work with the SAE, you enable either COPS-PR mode or COPS-XDR mode.

Highly Available Connections to JUNOSe Routers

JUNOSe routers maintain state information, a feature that allows an active, managing SAE to reconnect to a JUNOSe router without a performing a data resynchronization in the following instances:

- The network connection between the SAE and the JUNOSe router is disrupted, and the router reconnects to the SAE
- For JUNOSe routers with high availability configured, when the secondary SRP takes control from a failed SRP it can reconnect to the SAE

Adding JUNOSe Routers and Virtual Routers with the CLI

The SAE uses router and virtual router objects to manage interfaces on JUNOSe virtual routers. Each JUNOSe router in the SRC network and its virtual routers (VRs) must have a configuration.

There are two ways to add routers:

- Detect operative routers and configured JUNOSe VRs in the SRC network and add them to the configuration.
- Add each router and VR individually.

Adding Operative JUNOSe Routers and Virtual Routers

To add routers and JUNOSe VRs that are currently operative and have an operating SNMP agent:

- In operational mode, enter the following command:

  ```
  request network discovery network network <community community>
  ```

  where:

  - *network*—Address (with or without mask) of the network to discover
  - *community*—Name of the SNMP community to which the devices belong

If you add a router using the discover network feature, the software adds the IP address of the first SNMP agent on the router to respond to the discover request.

Adding Routers Individually

Use the following configuration statements to add a router:

```
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. This procedure uses junose_boston as the name of the router.

   ```
   user@host# edit shared network device junose_boston
   ```

2. (Optional) Add a description for the router.

   ```
   [edit shared network device junose_boston]
   user@host# set description description
   ```

3. (Optional) Add the IP address of the router.

   ```
   [edit shared network device junose_boston]
   user@host# set management-address management-address
   ```

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

   ```
   [edit shared network device junose_boston]
   user@host# set device-type junose
   ```

5. (Optional) Specify quality of service (QoS) profiles that are configured on the router.

   ```
   [edit shared network device junose_boston]
   user@host# set qos-profile [qos-profile...]
   ```

6. (Optional) Verify your configuration.

   ```
   [edit shared network device junose_boston]
   user@host# show
   description "Juniper Networks E320";
   management-address 10.10.8.27;
   device-type junose;
   qos-profile dhcp-default;
   interface-classifier {
     rule rule-0 {
       script #;
     }
   }
   ```

Adding Virtual Routers Individually

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. This procedure uses junose_Boston as the name of the router and vr1 as the name of the virtual router.

   ```
   user@host# edit shared network device junose_boston virtual-router vr1
   ```

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 junose_boston virtual-router vr1]
   user@host# set sae-connection [sae-connection...]
   ```

   To specify the active SAE and the redundant SAE, enter an exclamation point (!) after the hostname or IP address of the connected SAE. For example:

   ```
   [edit shared network device junose_boston virtual-router vr1]
   user@host# set sae-connection [sae1! sae2!]
   ```

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

   ```
   [edit shared network device junose_boston virtual-router vr1]
   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 junose_boston virtual-router vr1]
   user@host# **set snmp-write-community** *snmp-write-community*

5. (Optional) Specify service scopes assigned to this virtual router. The scopes are available for subscribers connected to this virtual router for selecting customized versions of services.

   [edit shared network device junose_boston virtual-router vr1]
   user@host# **set scope** [*scope*...]

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

   [edit shared network device junose_boston virtual-router vr1]
   user@host# **set local-address-pools** *local-address-pools*

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

   [edit shared network device junose_boston virtual-router vr1]
   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 junose_boston virtual-router vr1]
   user@host# **tracking-plug-in** [*tracking-plug-in*...]

9. (Optional) Verify your configuration.

   ```
   [edit shared network device junose_boston virtual-router vr1]
   user@host# show
   sae-connection 192.168.10.25;
     snmp-read-community ********;
     snmp-write-community ********;
     scope POP-Boston;
   local-address-pools "(10.25.8.0 10.25.20.255])";
   static-address-pools "({10.30.30.0/24,10.30.30.0,10.30.30.255})";
   tracking-plug-in flexRadius;
   ```

Related Topics

- For information about service scopes, see *SRC-PE Services and Policies Guide, Chapter 1, Managing Services with the SRC CLI*
- For information about local IP address pools, see Updating Local IP Address Pools for JUNOSe VRs on page 80.
- For information about tracking plug-ins, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 11, Configuring Accounting and Authentication Plug-Ins with the SRC CLI*

Configuring the SAE to Manage JUNOSe Routers with the CLI

To set up the SAE to manage JUNOSe routers, configure a router driver that specifies a COPS server that can accept COPS connections from the COPS client in JUNOSe routers.

Use the following configuration statements to configure the SAE to manage JUNOSe routers:

```
shared sae configuration driver junose {
    cops-server-port cops-server-port;
    backlog backlog;
    keepalive-interval keepalive-interval;
    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;
    pending-address-timeout pending-address-timeout;
    cops-handler-threads cops-handler-threads;
    cached-driver-expiration cached-driver-expiration;
    drop-unmanaged-interfaces-xdr-driver;
    track-unmanaged-interfaces-xdr-driver;
}
```

To configure the SAE to manage JUNOSe routers:

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

   ```
   user@host# edit shared sae group west-region configuration driver junose
   ```

2. Configure the port number of the SAE COPS server. The port number must match the configuration of the SRC client in the JUNOSe router.

   ```
   [edit shared sae group west-region configuration driver junose]
   user@host# set cops-server-port cops-server-port
   ```

3. Configure the number of outstanding connection attempts before connections are dropped.

   ```
   [edit shared sae group west-region configuration driver junose]
   user@host# set backlog backlog
   ```

4. Configure the interval between keepalive messages sent from the COPS client (the JUNOSe router).

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

5. Configure the timeout interval in which the COPS server waits for a response to COPS requests.

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

6. Configure the maximum length of a COPS message.

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

7. Configure the buffer size for receiving COPS messages from the JUNOSe client. We recommend that you use the default setting unless you are instructed to change it by Juniper Networks.

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

8. Configure the buffer size for sending COPS messages to the JUNOSe client. We recommend that you use the default setting unless you are instructed to change it by Juniper Networks.

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

9. Configure the maximum time that a DHCP address request remains pending.

   [edit shared sae group west-region configuration driver junose]
   user@host# **set pending-address-timeout** *pending-address-timeout*

10. Configure the size of the thread pool for handling unsolicited messages. These threads are shared among all JUNOSe router drivers.

    [edit shared sae group west-region configuration driver junose]
    user@host# **set cops-handler-threads** *cops-handler-threads*

11. Configure the minimum amount of time to keep the state of a router driver after its COPS connection has been closed.

    [edit shared sae group west-region configuration driver junose]
    user@host# **set cached-driver-expiration** *cached-driver-expiration*

12. (Optional) If you are using COPS-XDR, specify whether or not the JUNOSe router driver keeps a record of unmanaged interfaces.

    [edit shared sae group west-region configuration driver junose]
    user@host# **set drop-unmanaged-interfaces-xdr-driver**

13. (Optional) Enable or disable sending of interface-tracking events for unmanaged interfaces for the XDR router driver.

    [edit shared sae group west-region configuration driver junose]
    user@host# **set track-unmanaged-interfaces-xdr-driver**

14. (Optional) Verify your configuration.

    ```
    [edit shared sae group west-region configuration driver junose]
    user@host# show
    cops-server-port 3288;
    backlog 50;
    keepalive-interval 45;
    message-timeout 120000;
    cops-message-maximum-length 200000;
    ```

```
cops-message-read-buffer-size 30000;
cops-message-write-buffer-size 30000;
pending-address-timeout 5000;
cops-handler-threads 20;
cached-driver-expiration 600;
drop-unmanaged-interfaces-xdr-driver;
track-unmanaged-interfaces-xdr-driver;
```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*

Using SNMP to Retrieve Information from JUNOSe Routers

Some scripts in the SRC software use SNMP to get information from the router. For example, the **poolPublisher** router initialization script uses SNMP to read the IP pools.

- On the router, you can configure access to the router's SNMP server. See *Configuring the SNMP Server on the JUNOSe Router* on page 52.
- On the SAE, you can configure global default SNMP communities that are used for read and write access to the router. See *Configuring Global SNMP Communities in the SRC Software* on page 53.
- You can specify SNMP communities for each virtual router. We recommend that you specify communities for each virtual router instead of configuring global communities. See *Adding Virtual Routers Individually* on page 48.

Configuring the SNMP Server on the JUNOSe Router

Access to the SNMP server on the router by an SNMP client is governed by a proprietary SNMP community table. This table identifies communities that have read-only, read-write, or administrative permission to the SNMP Management Information Base (MIB) stored on a particular server.

When an SNMP server receives a request, the server extracts the client's IP address and the community name. The SNMP server searches the community table for a matching community.

- If a match is found, its access list name is used to validate the IP address.
    - If the access list name is null, the IP address is accepted.
    - If an invalid IP address results, an SNMP authentication error is sent to the SNMP client.
- If a match is not found, an SNMP authentication error results.

To configure the SNMP agent on the JUNOSe router:

1. Switch to the virtual router for which you want to create an SRC client.

   ```
   host1#(config)virtual-router <vrName>
   ```

2. Enable the SNMP agent.

   ```
   host1:<vrName>#(config)snmp-server
   ```

3. Configure at least one authorized SNMP read-write community (SNMPv1/v2c), which provides SNMP client access.

   ```
   host1:<vrName>(config)#snmp-server community boston rw
   ```

4. (Optional) Configure a read-only community.

   ```
   host1:<vrName>#(config)snmp-server public ro
   ```

Configuring Global SNMP Communities in the SRC Software

You can configure global default SNMP communities that are used if a VR does not exist on the router or if the community strings have not been configured for the VR.

Use the following configuration statements to configure global default SNMP communities:

```
shared sae configuration driver snmp {
    read-only-community-string read-only-community-string;
    read-write-community-string read-write-community-string;
}
```

To configure global default SNMP communities:

1. From configuration mode, access the configuration statements that configure default SNMP communities. In this sample procedure, the JUNOSe driver is configured in the west-region group.

   ```
   user@host# edit shared sae group west-region configuration driver snmp
   ```

2. Configure the default SNMP community string used for read access to the router.

   ```
   [edit shared sae group west-region configuration driver snmp]
   user@host# set read-only-community-string read-only-community-string
   ```

3. Configure the default SNMP community string used for write access to the router.

```
[edit shared sae group west-region configuration driver snmp]
user@host# set read-write-community-string read-write-community-string
```

4. (Optional) Verify your configuration.

```
[edit shared sae group west-region configuration driver snmp]
user@host# show
read-only-community-string ********;
read-write-community-string ********;
```

Developing Router Initialization Scripts

When the SAE establishes a connection with a router, it can run a router initialization script to customize the setup of the connection. Router initialization scripts are run when the connection between a router and the SAE is established and again when the connection is dropped.

For JUNOSe VRs that supply IP addresses from a local pool, a router initialization script is provided that identifies which VR supplies each IP pool and writes the information to the configuration. The SAE runs the script only when a COPS connection is established to the JUNOSe router. Consequently, if you modify information about IP pools on a VR after the COPS connection is established, the SAE will not automatically register the changes, and you must update the configuration.

Table 4 describes the router initialization scripts that we provide with the SRC software in the */opt/UMC/sae/lib* folder.

Table 4: Router Initialization Scripts

Script Name	Function	When to Use Script
iorPublisher	Publishes the IOR of the SAE into an internal part of the shared configuration so that a NIC can associate a router with an SAE.	Use with JUNOSe routers that do not supply IP addresses from local pools, and with JUNOS routing platforms.
poolPublisher	Publishes the IOR of the SAE and local IP address pools in the directory so that a NIC can associate a router with an SAE and resolve the IP-to-SAE mapping.	Use with JUNOSe virtual routers that supply IP addresses from local pools.

Interface Object Fields

Router initialization scripts are written in the Python programming language (www.python.org) and executed in the Jython environment (www.jython.org).

Router initialization scripts interact with the SAE through an interface object called Ssp. The SAE exports a number of fields through the interface object to the script and expects the script to provide the entry point to the SAE.

Table 5 describes the fields that the SAE exports.

Table 5: Exported Fields

Ssp Attribute	Description
Ssp.properties	System properties object (class: java.util.Properties)—The properties should be treated as read-only by the script.
Ssp.errorLog	Error logger—Use the SsperrorLog.printIn (message) to send error messages to the log.
Ssp.infoLog	Info logger—Use the Ssp.infoLog.printIn (message) to send informational messages to the log.
Ssp.debugLog	Debug logger—Use the Ssp.debugLog.printIn (message) to send debug messages to the log.

The router initialization script must set the field Ssp.routerInit to a factory function that instantiates a router initialization object:

- <VRName>—Name of the virtual router in which the COPS client has been configured, format: virtualRouterName@RouterName
- <virtualIp>—Virtual IP address of the SAE (string, dotted decimal; for example: 192.168.254.1)
- <realIp>—Real IP address of the SAE (string, dotted decimal; for example, 192.168.1.20)
- <VRIp>—IP address of the virtual router (string, dotted decimal)
- <transportVR>—Name of the virtual router used for routing the COPS connection, or None, if the COPS client is directly connected

The factory function must implement the following interface:

```
Ssp.routerInit(VRName,
          virtualIp,
          realIp,
          VRIp,
          transportVR)
```

The factory function returns an interface object that is used to set up and tear down a connection for a given COPS server. A common case of a factory function is the constructor of a class.

The factory function is called directly after a COPS server connection is established. In case of problems, an exception should be raised that leads to the termination of the COPS connection.

Required Methods

Instances of the interface object must implement the following methods:

- *setup()*—Is called when the COPS server connection is established and is operational. In case of problems, an exception should be raised that leads to the termination of the COPS connection.
- *shutdown()*—Is called when the COPS server connection to the virtual router is terminated. This method should not raise any exceptions in case of problems.

Example: Router Initialization Script

The following script defines a router initialization class named *SillyRouterInit*. The interface class does not implement any useful functionality. The interface class just writes messages to the infoLog when the router connection is created or terminated.

```
class SillyRouterInit:
    def __init__(self, vrName, virtualIp, realIp, vrIp, transportVr):
        """ initialize router initialization object """
        self.vrName = vrName
        Ssp.infoLog.printin("SillyRouterInit created")

    def setup(self):
        """ initialize connection to router """
        Ssp.infoLog.printin("Setup connection to VR %(vrName)s" %
                    vars(self))

    def shutdown(self):
        """ shutdown connection to router """
        Ssp.infoLog.printin("Shutdown connection to VR %(vrName)s" %
                    vars(self))

#
# publish interface object to Ssp core
#
Ssp.routerInit = SillyRouterInit
```

Specifying Router Initialization Scripts on the SAE with the CLI

Use the following configuration statements to specify router initialization scripts for JUNOSe routers:

```
shared sae configuration driver scripts {
    extension-path extension-path;
    general general;
    junose-pr junose-pr;
    junose-xdr junose-xdr;
    }
```

To configure router initialization scripts for JUNOSe routers:

1. From configuration mode, access the configuration statements that configure router initialization scripts. In this sample procedure, the scripts are configured in the west-region group.

   ```
   user@host# edit shared sae group west-region configuration driver scripts
   ```

2. Specify the script for JUNOSe routers when the JUNOSe driver uses COPS-PR mode when connecting to the SAE.

   ```
   [edit shared sae group west-region configuration driver scripts]
   user@host# set junose-pr junose-pr
   ```

3. Specify the script for JUNOSe routers when the JUNOSe driver uses COPS-XDR mode when connecting to the SAE.

   ```
   [edit shared sae group west-region configuration driver scripts]
   user@host# set junose-xdr junose-xdr
   ```

   In COPS-XDR mode, the router does not send the network access server (NAS) IP address to the SAE. If your configuration requires this value, add the following line to a JUNOSe script:

   import ERXnasip

   When you add the **import ERXnasip** entry, the script obtains the NAS-IP address from the router through SNMP. This mechanism can affect performance, especially when the SAE manages a large number of virtual routers.

4. Configure a router initialization script that can be used for all types of routers that the SRC software supports.

   ```
   [edit shared sae group west-region configuration driver scripts]
   user@host# set general general
   ```

5. Configure a path to router initialization scripts that are not in the default location, */opt/UMC/sae/lib*.

   ```
   [edit shared sae group west-region configuration driver scripts]
   user@host# set extension-path extension-path
   ```

6. (Optional) Verify your router initialization script configuration.

   ```
   [edit shared sae group west-region configuration driver scripts]
   user@host# show
   junose-xdr poolPublisher;
   ```

Accessing the Router CLI

You can access the CLIs of Juniper Networks routers through a Telnet or secure shell connection.

- To open a Telnet session to a router, use the telnet operational mode command. For example:

  ```
  user@host> telnet 10.10.10.3
  ```

- To open a secure shell connection, use the ssh operational command. For example:

  ```
  user@host> ssh host 10.10.10.3
  ```

Starting the SRC Client on a JUNOSe Router

JUNOSe routers use an SRC client to interact with the SAE. See *JUNOSe Broadband Access Configuration Guide* for complete information about configuring the SRC client on a JUNOSe router.

To start the SRC client:

1. Access the router CLI.
2. Access Global configuration mode.

   ```
   host1#configure terminal
   ```

3. Switch to the virtual router for which you want to create an SRC client.

   ```
   host1(config)#virtual-router <vrName>
   ```

4. Enable the SRC client.

   To enable COPS-PR mode:

   ```
   host1:<vrName>(config)#sscc enable cops-pr
   ```

   To enable COPS-XDR mode:

   ```
   host1:<vrName>(config)#sscc enable
   ```

5. Set the primary address from the configuration directory.

   ```
   host1:<vrName>(config)#sscc primary address <ipAddress> port 3288
   ```

Stopping the SRC Client on a JUNOSe Router

JUNOSe routers use an SRC client to interact with the SAE. See *JUNOSe Broadband Access Configuration Guide* for complete information about configuring the SRC client on the JUNOSe router.

To stop the SRC client:

1. Access the router CLI.

   See *Accessing the Router CLI* on page 58.

2. Access Global configuration mode.

   ```
   host1#configure terminal
   ```

3. Switch to the virtual router for which you want to stop an SRC client.

   ```
   host1(config)#virtual-router <vrName>
   ```

4. Disable the SRC client.

   ```
   host1:<vrName>(config)#no sscc enable
   ```

Monitoring Interactions Between the SAE and the JUNOSe Router

To monitor the connection between the router and the SAE:

- Use the `show sscc info` command on the JUNOSe router

To display the version number of the SRC client:

- Use the `show sscc version` command on the JUNOSe router.

See the *JUNOSe Command Reference Guide* for details about these commands.

You can also monitor the interactions between the SRC software and the router in the log files for the SAE and in the log files generated by the JUNOSe router.

- For information about configuring logging for the SAE, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 3, Configuring Logging for SRC Components with the CLI*.
- For information about configuring logging on JUNOSe routers, see *JUNOSe System Event Logging Reference Guide.*

Troubleshooting Problems with Managing JUNOSe Routers

You can troubleshoot problems with the SRC client on JUNOSe routers and with managed JUNOSe routers, interfaces, and services on the SAE.

Troubleshooting the SRC Client on JUNOSe Routers

To troubleshoot SRC problems on the router:

1. Look at the log files for the SAE and the log files generated by the SRC client on the JUNOSe router.
    - If the log files indicate a problem with specific interfaces on the router, review the configuration of the associated policies in the SRC software, and fix any errors.
    - If the log files indicate a problem with a specific service or its associated policy rules, review the configuration of the service or policies in the SRC software, and fix any errors.
    - If the log files indicate only that the SRC client is not responding, ensure that the values in the SAE configuration match the values in the SRC client configuration on the router.
2. Restart the SRC client on the JUNOSe router.

    When you restart the SRC client, the SRC client removes all policies that were installed by the SRC software and reports all interfaces again.

NOTE: DHCP addresses that were managed are not reported again, so we recommend that you do not restart the SRC client if you are managing DHCP sessions.

To restart the SRC client in COPS-PR mode, enter the following commands:

```
host1:<vrName>(config)#no sscc enable
host1:<vrName>(config)#sscc enable cops-pr
```

To restart the SRC client in COPS-XDR mode, enter the following commands:

```
host1:<vrName>(config)#no sscc enable
host1:<vrName>(config)#sscc enable
```

If restarting the SRC client does not resolve the problem, rebuild the router configuration and restart the client.

Viewing the State of JUNOSe Device Drivers with the SRC CLI

To display the state of JUNOSe drivers, use the following operational mode command.

show sae drivers <device-name *device-name*> < (brief) > <maximum-results *maximum-results*>

For example:

```
user@host> show sae drivers device-name default@dryad
JUNOSe Driver
Device name                         default@dryad
Device type                         junose
Device IP                           10.227.7.244
Local IP                            10.227.7.172
TransportRouter                     default@dryad
Device version                      7.2.0
Start time                          Tue Feb 13 14:18:44 EST 2007
Number of notifications             20
Number of processed added           14
Number of processed changed         0
Number of processed deleted         6
Number of provisioning attempt      30
Number of provisioning attempt failed 0
Number of outstanding decisions     0
Number of SAP                       7
Number of PAP                       1

  Job Queue
  Size                 0
  Age (ms)             1
  Total enqueued       28
  Total dequeued       28
  Average job time (ms) 426

  State Synchronization
  Number recovered subscriber sessions             0
  Number recovered service sessions                0
  Number recovered interface sessions              0
  Number invalid subscriber sessions               0
  Number invalid service sessions                  0
  Number invalid interface sessions                0
  Background restoration start time                Tue Feb 13 14:18:49 EST 2007
  Background restoration end time                  Tue Feb 13 14:18:49 EST 2007
  Number subscriber sessions restored in background 0
  Number of provisioning objects left to collect    0
  Total number of provisioning objects to collect   11
  Start time                                       Tue Feb 13 14:18:45 EST 2007
  End time                                         Tue Feb 13 14:18:47 EST 2007
  Number of synched contexts                       7
  Number of post-sync jobs                         6
```

Viewing Statistics for Specific JUNOSe Device Drivers with the SRC CLI

To display statistics for a specific JUNOSe device driver, use the following operational mode command:

show sae statistics device <name *name*> < (brief) >

For example:

```
user@host> show sae statistics device name default@dryad
SNMP Statistics
Add notification handle time       6
Change notification handle time    0
Client ID                          default@dryad
Delete notification handle time    0
Failover IP                        0.0.0.0
Failover port                      0
Handle message time                60
Job queue age                      0
Job queue time                     4
Number message send                158
Number of added jobs               9
Number of add notifications        4
Number of change notifications     0
Number of delele notifications     0
Number of managed interfaces       4
Number of message errors           0
Number of message timeouts         0
Number of removed jobs             9
Number of user session established 0
Number of user session removed     0
Router type                        JUNOSE COPS
Up time                            172286
Using failover server              false
```

Viewing Statistics for All JUNOSe Device Drivers with the SRC CLI

To display SNMP statistics for all JUNOSe device drivers, use the following operational mode command:

show sae statistics device common junose-cops

For example:

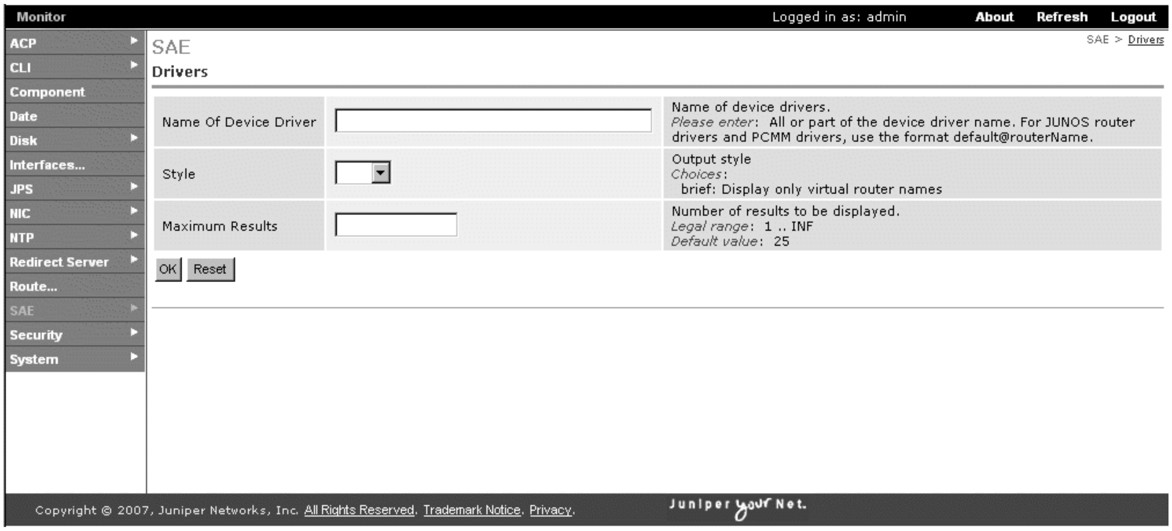
```
user@host> show sae statistics device common junose-cops
SNMP Statistics
Driver type                    JUNOSE COPS
Number of close requests       0
Number of connections accepted 2
Number of current connections  1
Number of open requests        2
Server address                 0:0:0:0:0:0:0:0
Server port                    3288
Time since last redirect       186703
```

Viewing the State of JUNOSe Device Drivers with the C-Web Interface

If the log files indicate a problem with a specific driver, review the configuration of the associated with the JUNOSe router driver with the C-Web interface.

1. Select **SAE** from the side pane, and click **Drivers**.

   The Drivers pane appears.



SAE

Drivers

Name Of Device Driver		Name of device drivers. *Please enter*: All or part of the device driver name. For JUNOS router drivers and PCMM drivers, use the format default@routerName.
Style		Output style *Choices*: brief: Display only virtual router names
Maximum Results		Number of results to be displayed. *Legal range*: 1 .. INF *Default value*: 25

2. In the Name of Device Driver box, enter a full or partial device driver name for which you want to display information, or leave the box blank to display all devices. Use the format:

   <virtual router name>@<router name>

3. Select an output style from the Style list.

4. In the Maximum Results box, enter the maximum number of results that you want to receive.

5. Click **OK**.

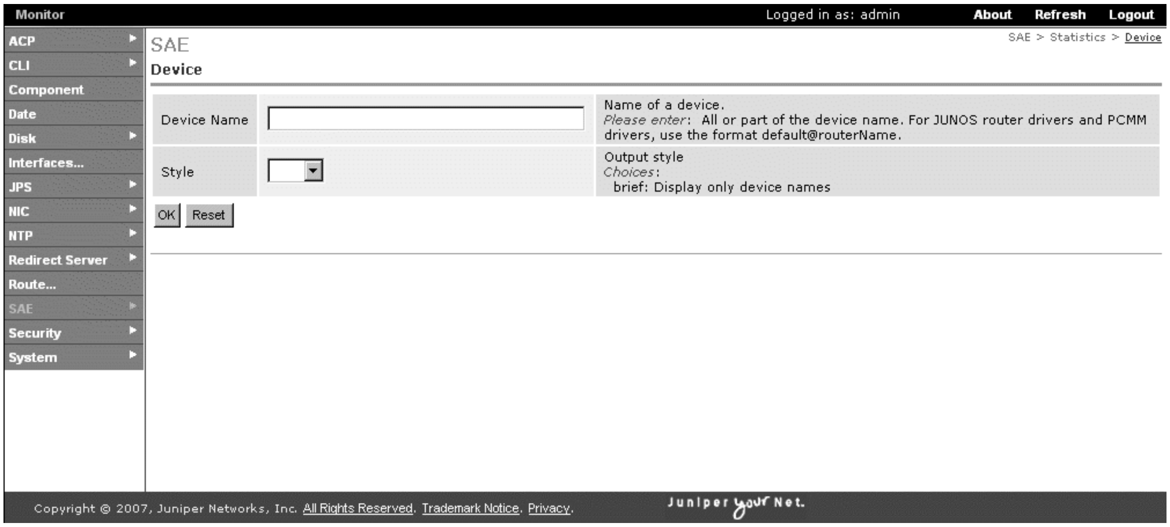
   The Drivers pane displays information about the JUNOSe device driver.

Viewing Statistics for Specific JUNOSe Device Drivers with the C-Web Interface

To view SNMP statistics about devices:

1. Select **SAE** from the side pane, click **Statistics**, and then click **Device**.

   The Device pane appears.



SAE

Device

Device Name		Name of a device. *Please enter*: All or part of the device name. For JUNOS router drivers and PCMM drivers, use the format default@routerName.
Style		Output style *Choices*: brief: Display only device names

2. In the Device Name box, enter a full or partial device name for which you want to display information, or leave the box blank to display all devices.

3. Select an output style from the Style list.

4. Click **OK**.

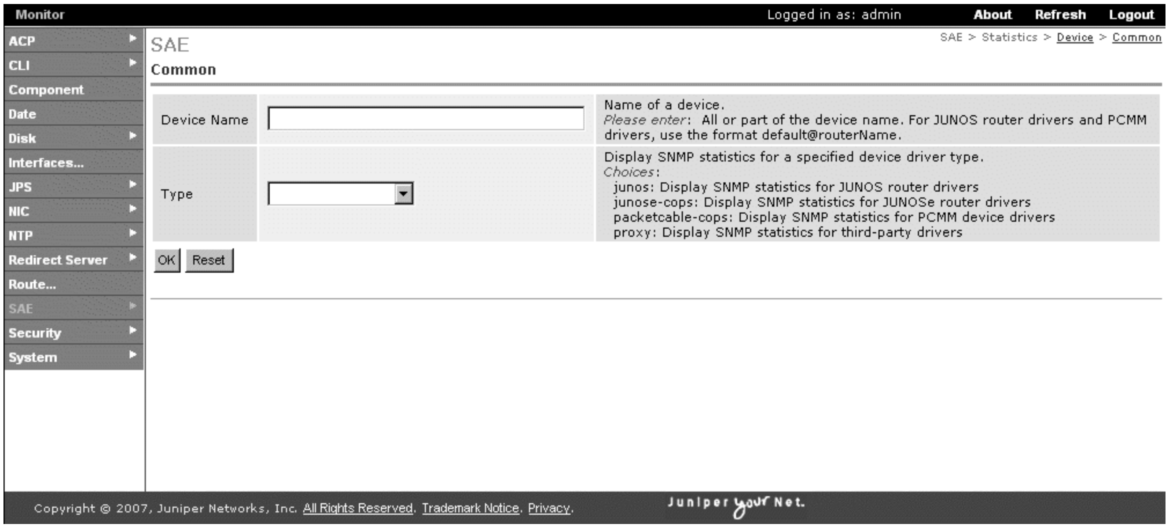
   The Device pane displays statistics for all devices.

Viewing Statistics for All JUNOSe Device Drivers with the C-Web Interface

To view SNMP statistics about specific devices:

1. Select **SAE** from the side pane, click **Statistics**, click **Device**, and then click **Common**.

   The Common pane appears.



Name of a device.
Please enter: All or part of the device name. For JUNOS router drivers and PCMM drivers, use the format default@routerName.

Display SNMP statistics for a specified device driver type.
Choices:
junos: Display SNMP statistics for JUNOS router drivers
junose-cops: Display SNMP statistics for JUNOSe router drivers
packetcable-cops: Display SNMP statistics for PCMM device drivers
proxy: Display SNMP statistics for third-party drivers

2. In the Device Name box, enter a full or partial device name for which you want to display information, or leave the box blank to display all devices.

3. Select **junose-cops** from the Type list.

4. Click **OK**.

   The Common pane displays statistics for the specified device.

Chapter 6

Using JUNOSe Routers in the SRC Network with a Solaris Platform

This chapter describes how to set up the SRC software on a Solaris platform with the SRC configuration applications that run only on Solaris platforms. It also shows how to set up a JUNOSe router so that the router can be used the SRC network. It includes information about how to monitor the interactions between the SAE and the JUNOSe router and how to troubleshoot SRC problems on the router.

You can also use the following to configure JUNOSe routers:

- To use the SRC CLI, see *Chapter 5, Using JUNOSe Routers in the SRC Network with the SRC CLI*.
- To use the C-Web interface, see *SRC-PE C-Web Interface Configuration Guide, Chapter 17, Using JUNOSe Routers in the SRC Network with the C-Web Interface*.

Topics in this chapter include:

COPS Connection Between JUNOSe Routers and the SAE

Configuring the SRC client on a JUNOSe router opens a Common Open Policy Service (COPS) protocol layer connection to the SAE. When the SRC client software establishes a TCP/IP connection to the SAE, the SAE starts to manage the JUNOSe router. Subsequently, the SRC client sends configuration changes made on the JUNOSe router to the SAE, and the SAE updates SRC configurations for services and policies accordingly.

The SAE supports two versions of COPS:

- COPS usage for policy provisioning (COPS-PR)
- COPS External Data Representation Stand (COPS XDR)

The version of COPS that you use depends on the version of COPS that your JUNOSe router supports. When you set up your JUNOSe router to work with the SAE, you enable either COPS-PR mode or COPS XDR mode.

Highly Available Connections to JUNOSe Routers

JUNOSe routers maintain state information, a feature that allows an active, managing SAE to reconnect to a JUNOSe router without a performing a data resynchronization in the following instances:

- The network connection between the SAE and the JUNOSe router is disrupted, and the router reconnects to the SAE
- For JUNOSe routers with high availability configured, when the secondary SRP takes control from a failed SRP it can reconnect to the SAE

Adding JUNOSe Routers and Virtual Routers

The SAE uses router and virtual router objects in the directory to manage interfaces on JUNOSe virtual routers. Each JUNOSe router in the SRC network and its virtual routers (VRs) must appear in the directory. There are two ways to add routers to the directory:

- Use SDX Admin to detect operative routers and configured JUNOSe VRs in the SRC network and add them to the directory.
- Add each router and VR individually. You need to add routers and VRs individually if you use an LDAP client other than SDX Admin or if you want to add inoperative routers or unconfigured JUNOSe VRs.

NOTE: You must define connected SAEs for each router in the virtual router object of the directory. This step is required for the SAE to work with the router. See *Specifying the SAEs That Can Manage the Router* on page 75.

Adding Operative JUNOSe Routers and Virtual Routers

To simultaneously add to the directory routers and JUNOSe VRs that are currently operative and have an operating Simple Network Management Protocol (SNMP) agent:

1. In the SDX Admin navigation pane, select **o = Network**, and right-click.
2. Select **Discover Network**.

   The Discover Network dialog box appears.
3. Enter the IP address, the prefix of the network, and the SNMP community string.
4. Click **OK**.

   Objects for all routers and JUNOSe VRs that meet the criteria you specified appear under the Networks object in the navigation pane. You can modify the configuration of these objects. For information about configuring these objects, see *Adding Routers Individually* on page 69 and *Adding Virtual Routers Individually* on page 71.

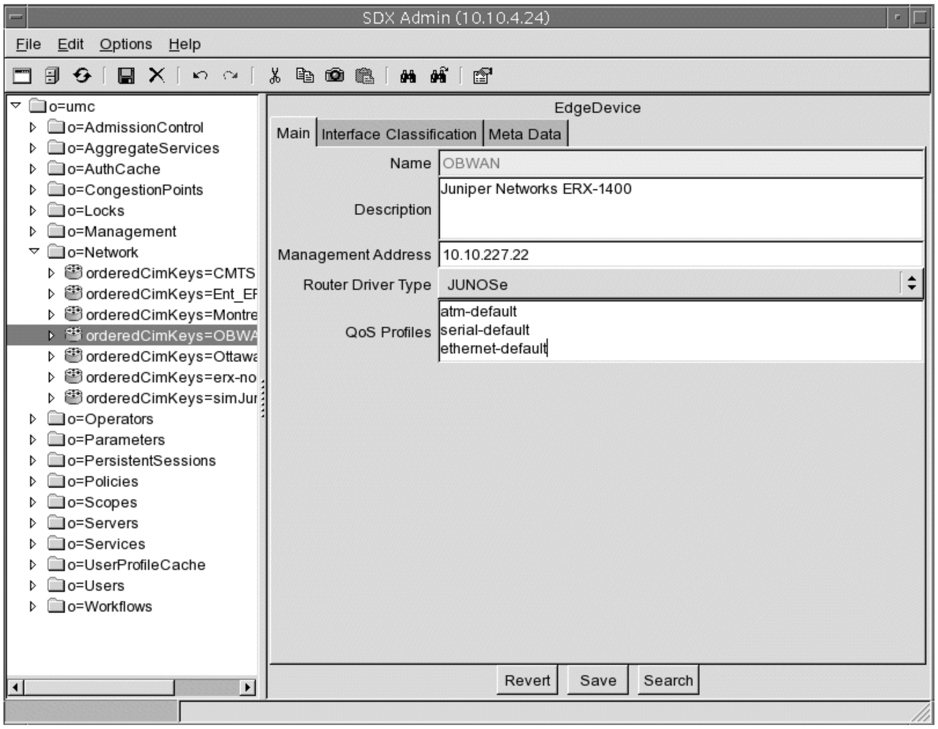
Adding Routers Individually

To add a single router to the directory with SDX Admin:

1. In the navigation pane, right-click the **Network** folder, and select **New > EdgeDevice**.

   The New EdgeDevice dialog box appears.
2. Enter the name of the router exactly as it is configured on the router, and click **OK**.

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



Name: OBWAN

Description: Juniper Networks ERX-1400

Management Address: 10.10.227.22

Router Driver Type: JUNOSe

QoS Profiles:
atm-default
serial-default
ethernet-default

3. In the content pane, edit or accept the default values for the router fields.

   See *Router Fields* on page 70.

4. Click **Save**.

Router Fields

In SDX Admin, you can modify the following fields in the content pane for a router (*orderedCimKeys = <EdgeDeviceName>, o = network, o = umc)*.

Description

- Information about this device; keywords that the find utility uses.
- Value—Text string
- Example—ERX-1400 router located in Ottawa

Management Address

- IP address of the router. If you add a router using the discover network feature, the software automatically adds the IP address of the first SNMP agent on the router to respond to the discover request.
- Value—IP address
- Example—192.0.1.1

Router Driver Type

- Type of device that this router object will be used to manage.
- Value
  - JUNOSe—JUNOSe router
  - JUNOS—JUNOS routing platform
  - PCMM—CMTS device
- Default—No value

QoS Profiles

- For JUNOSe routers, specifies quality of service (QoS) profiles that are configured on the router. To update this list, see *SRC-PE Solutions Guide, Chapter 1, Managing Tiered and Premium Services with QoS on JUNOSe Routers with the SRC CLI*.
- Value—List of QoS profiles on separate lines
- Guideline—This field applies to JUNOSe routers only
- Example—atm-default

Adding Virtual Routers Individually

To add a VR to the directory with SDX Admin:

1. In the navigation pane, right-click the device to which you want to add the VR, and select **New > VirtualRouter**.

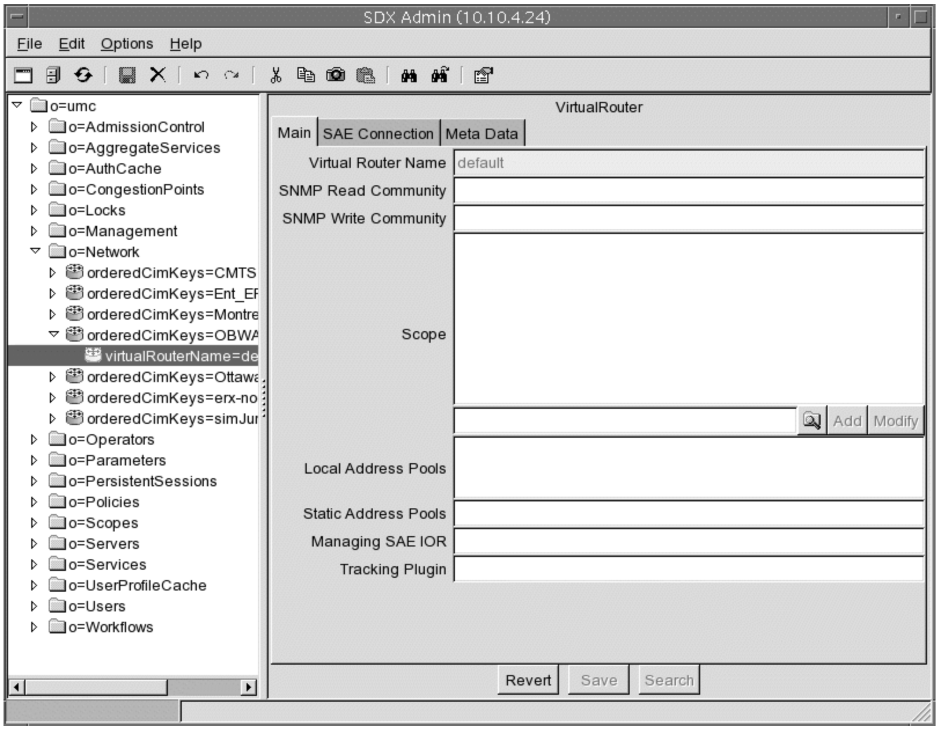
   The New VirtualRouter dialog box appears.

2. Enter the name of the VR, and click **OK**.

   - For JUNOSe routers, the name of the VR, which is case sensitive, must exactly match the name of the VR configured on the router.

   - For JUNOS routing platforms and CMTS devices, use the name default.

   The name of the new VR appears in the navigation pane, and the VirtualRouter pane appears.



Virtual Router Name: default

3. In the Main tab in the VirtualRouter pane, edit or accept the default values for the fields.

   See *Virtual Router Fields* on page 72.

4. Select the **SAE Connection** tab in the VirtualRouter pane, and add SAEs that are connected to the router.

   See *Specifying the SAEs That Can Manage the Router* on page 75.

NOTE: This step is required for the SAE to work with the router.

5. Click **Save**.

Virtual Router Fields

In SDX Admin, you can modify the following fields in the content pane for a virtual router (*virtualRouterName = <virtualRouterName = <name of virtual router> orderedCimKeys = <EdgeDeviceName>, o = network, o = umc)*.

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.
- Value—Text string
- Example—POP-Westford

Local Address Pools

- List of IP address pools that a JUNOSe VR currently manages and stores.
- Value—You can specify an unlimited number of ranges of local IP address pools for JUNOSe VRs. You can specify either the first and last addresses in a range or the first IP address and a factor that indicates the start of the range. You can also specify IP addresses to exclude. Use spaces in the syntax only to separate the first and last explicit IP addresses in a range.

  The IP pool syntax has the 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>—IP address mask
  - <digitNumber>—Integer specifying the number of significant digits of the first IP address in the range
  - <ipAddressExclude>—List of IP addresses to be excluded from the range
  - |—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
- Guidelines—If you do not configure the **PoolPublisher** router initialization scripts for a JUNOSe router, configure this field for the JUNOSe VR.
- Default—No value
- Example—This example shows four ranges for the IP address pool.

  ```
  ([10.10.10.5 10.10.10.250]
  {10.20.20.0/24}
  {10.21.0.0/255.255.0.0}
  ```

{10.20.30.0/24,10.20.30.1})

- The first range (a simple range) specifies all the IP addresses between the two IP addresses 10.10.10.5 and 10.10.10.250.
- The second range specifies all the IP addresses in the range 10.20.20.0 to 10.20.20.255.
- The third range uses a network mask to specify all the IP addresses in the range 10.21.0.0 to 10.21.255.255.
- The fourth range specifies all the addresses of the network 10.20.30.0 to 10.20.30.255, excluding the address 10.20.30.1.

Static Address Pools

- List of IP address pools that a JUNOSe VR manages but does not store. You can configure these address pools only in the SRC software.
- Value—See the field Local Address Pools.
- Guidelines—Configure this field on JUNOSe and CMTS VRs only.
- 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.
- Default—No value
- Guidelines—The **PoolPublisher** and **IorPublisher** router initialization scripts provide this information when the router connects to the SAE. If you do not use one of these router initialization scripts, enter a value in this field.
- Example—One of the following items:
  - Absolute path—*/opt/UMC/sae/var/run/sae.ior*
  - corbaloc URL—corbaloc::boston:8801/SAE
  - Actual IOR—IOR:000000000000002438444C3A736D67742E6A756E697...

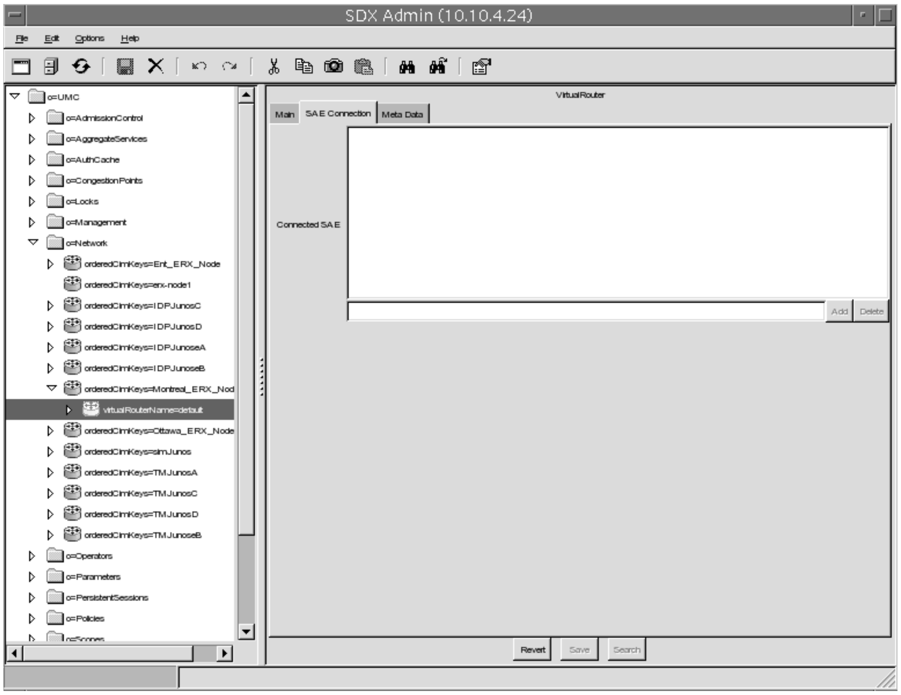
Tracking Plug-in

- Plug-ins that track interfaces that the SAE manages on this VR. The SAE calls these plug-in instances for every interface it manages. 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 network information collector (NIC) SAE plug-in agents that are specific to this VR.
- Default—No value
- Example—nicsae, flexRadius

Specifying the SAEs That Can Manage the Router

You must add the addresses of SAEs that can manage this router. This step is required for the SAE to work with the router. To add the SAEs, select the SAE Connection tab in the VirtualRouter pane.



Adding an SAE

To add an SAE:

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

   To specify the active SAE and the redundant SAE, enter an exclamation point (!) after the hostname or IP address of the connected SAE.

2. Click **Add**.

Modifying an SAE Address

To modify an SAE address:

1. 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**.

Deleting an SAE Address

To delete an SAE address:

1. 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 router or CMTS device.
- Value—IP addresses
- Default—No value

Configuring the SAE to Manage JUNOSe Routers

To set up the SAE to manage JUNOSe routers, you need to configure a router driver that specifies the COPS connection between the SAE COPS server and the COPS client in the JUNOSe router. See *Chapter 5, Using JUNOSe Routers in the SRC Network with the SRC CLI*.

Using SNMP to Retrieve Information from JUNOSe Routers

Some scripts in the SRC software use SNMP to get information from the router. For example, the **poolPublisher** router initialization script uses SNMP to read the IP pools.

- On the router, you can configure access to the router's SNMP server. See *Configuring the SNMP Server on the JUNOSe Router* on page 77.
- On the SAE, you can configure global default SNMP communities that are used for read and write access to the router. See *Configuring Global SNMP Communities in the SRC Software* on page 77.
- In the directory, you can specify SNMP communities for each virtual router. We recommend that you specify communities for each virtual router instead of global communities. See *Adding Virtual Routers Individually* on page 71.

Configuring the SNMP Server on the JUNOSe Router

Access to the SNMP server on the router by an SNMP client is governed by a proprietary SNMP community table. This table identifies communities that have read-only, read-write, or administrative permission to the SNMP Management Information Base (MIB) stored on a particular server.

When an SNMP server receives a request, the server extracts the client's IP address and the community name. The SNMP server searches the community table for a matching community.

- If a match is found, its access list name is used to validate the IP address.
  - If the access list name is null, the IP address is accepted.
  - If an invalid IP address results, an SNMP authentication error is sent to the SNMP client.
- If a match is not found, an SNMP authentication error results.

To configure the SNMP agent on the JUNOSe router:

1. Switch to the virtual router for which you want to create an SRC client.

   ```
   host1#(config)virtual-router <vrName>
   ```

2. Enable the SNMP agent.

   ```
   host1:<vrName>#(config)snmp-server
   ```

3. Configure at least one authorized SNMP read-write community (SNMPv1/v2c), which provides SNMP client access.

   ```
   host1:<vrName>(config)#snmp-server community boston rw
   ```

4. (Optional) Configure a read-only community.

   ```
   host1:<vrName>#(config)snmp-server public ro
   ```

Configuring Global SNMP Communities in the SRC Software

You can configure global default SNMP communities that are used if a VR does not exist on the router or the community strings have not been configured for the VR. See *Chapter 5, Using JUNOSe Routers in the SRC Network with the SRC CLI*.

Developing Router Initialization Scripts

When the SAE establishes a connection with a router, it can run a router initialization script to customize the setup of the connection. Router initialization scripts are run when the connection between a router and the SAE is established and again when the connection is dropped.

For JUNOSe VRs that supply IP addresses from a local pool, a router initialization script is provided that identifies which VR supplies each IP pool and writes the information to the directory. The SAE runs the script only when a COPS connection is established to the JUNOSe router. Consequently, if you modify information about IP pools on a VR after the COPS connection is established, the SAE will not automatically register the changes, and you must update the directory.

Table 6 describes the router initialization scripts that we provide with the SRC software in the */opt/UMC/sae/lib* directory.

Table 6: Router Initialization Scripts

Script Name	Function	When to Use Script
IorPublisher	Publishes the IOR of the SAE in the directory so that a NIC can associate a router with an SAE.	Use with JUNOSe routers that do not supply IP addresses from local pools, and JUNOS routing platforms.
poolPublisher	Publishes the IOR of the SAE and local IP address pools in the directory so that a NIC can associate a router with an SAE and resolve the IP-to-SAE mapping.	Use with JUNOSe virtual routers that supply IP addresses from local pools.

Interface Object Fields

Router initialization scripts interact with the SAE through an interface object called Ssp. The SAE exports a number of fields through the interface object to the script and expects the script to provide the entry point to the SAE.

Table 7 describes the fields that the SAE exports.

Table 7: Exported Fields

Ssp Attribute	Description
Ssp.properties	System properties object (class: java.util.Properties)—The properties should be treated as read-only by the script.
Ssp.errorLog	Error logger—Use the SsperrorLog.printIn (message) to send error messages to the log.
Ssp.infoLog	Info logger—Use the Ssp.infoLog.printIn (message) to send informational messages to the log.
Ssp.debugLog	Debug logger—Use the Ssp.debugLog.printIn (message) to send debug messages to the log.

The router initialization script must set the field Ssp.routerInit to a factory function that instantiates a router initialization object:

- < VRName > —Name of the virtual router in which the COPS client has been configured, format: virtualRouterName@RouterName
- < virtualIp > —Virtual IP address of the SAE (string, dotted decimal; for example: 192.168.254.1)
- < realIp > —Real IP address of the SAE (string, dotted decimal; for example, 192.168.1.20)

- <VRIp>—IP address of the virtual router (string, dotted decimal)
- <transportVR>—Name of the virtual router used for routing the COPS connection, or None, if the COPS client is directly connected

The factory function must implement the following interface:

Ssp.routerInit(VRName,
virtualIp,
realIp,
VRIp,
transportVR)

The factory function returns an interface object that is used to set up and tear down a connection for a given COPS server. A common case of a factory function is the constructor of a class.

The factory function is called directly after a COPS server connection is established. In case of problems, an exception should be raised that leads to the termination of the COPS connection.

Required Methods

Instances of the interface object must implement the following methods:

- *setup()*—Is called when the COPS server connection is established and is operational. In case of problems, an exception should be raised that leads to the termination of the COPS connection.
- *shutdown()*—Is called when the COPS server connection is terminated to the virtual router. This method should not raise any exceptions in case of problems.

Example: Router Initialization Script

The following script defines a router initialization class named *SillyRouterInit*. The interface class does not implement any useful functionality. The interface class just writes messages to the infoLog when the router connection is created or terminated.

```
class SillyRouterInit:
    def __init__(self, vrName, virtualIp, realIp, vrIp, transportVr):
        """ initialize router initialization object """
        self.vrName = vrName
        Ssp.infoLog.printin("SillyRouterInit created")

    def setup(self):
        """ initialize connection to router """
        Ssp.infoLog.printin("Setup connection to VR %(vrName)s" %
                    vars(self))

    def shutdown(self):
        """ shutdown connection to router """
        Ssp.infoLog.printin("Shutdown connection to VR %(vrName)s" %
                    vars(self))
```

```
#
# publish interface object to Ssp core
#
Ssp.routerInit = SillyRouterInit
```

Specifying Router Initialization Scripts on the SAE

To specify router initialization scripts, see *Chapter 5, Using JUNOSe Routers in the SRC Network with the SRC CLI*.

Updating Local IP Address Pools for JUNOSe VRs

When you reconfigure local IP address pools on a JUNOSe VR, you must update in the directory the local IP addresses that the VR provides.

Before you update local IP address pools, make sure that:

- The JUNOSe router and VR appear in the directory.
- The VR has an operating SNMP agent.
- The host that supports SDX Admin or the SAE can communicate with the VR through SNMP.
- You have write permissions for the *o = Network* subtree.

There are two ways to add routers to the directory:

- SDX Admin—Updates on VR at a time.
- The **poolRepublish** command—simultaneously updates any number of VRs in the same directory.

Updating Local IP Address Pools with SDX Admin

To allow updates of IP address pools with SDX Admin, the host that supports SDX Admin must be able to communicate with the VR through SNMP. To update local IP address pools for a VR in the directory with SDX Admin:

1. In the navigation pane, expand **o = Network**.
2. In the navigation pane, expand the object for the router on which the VR is configured.
3. Right-click the object for the VR in the navigation pane.
4. Select **Update IP Pools**.

   The SDX Admin dialog box appears.
5. Enter the IP address for the VR, enter the SNMP community if the default value is incorrect, and click **OK**.

SDX Admin updates the local IP addresses for the VR in the directory and displays the information in the Local IP Address field of the Main tab in the VirtualRouter pane.

Updating Local IP Address Pools with the poolRepublish Command

You can use the **poolRepublish** command on the SAE host to update local IP address pools. You can specify multiple VRs with the **poolRepublish** command that use the same SNMP read community. For each VR you must specify the name of the VR, the name of the JUNOSe router on which it is configured, the VR's corresponding IP address, and the directory connection.

To update local IP addresses using the **poolRepublish** command:

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

   ```
   cd /opt/UMC/sae/etc
   ```

2. Run the command.

   ```
   ./poolRepublish -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
   ```

   The software updates and displays the local IP address pools for each VR you specified.

   ```
   vr1@erx1 pools: ([10.227.11.242 10.227.11.250][10.227.11.226
   10.227.11.239]{10.227.11.208/255.255.255.240}{10.227.11.240/255.255.2
   55.240}{10.227.11.224/255.255.255.240})
   vr2@erx2: ([10.227.12.242 10.227.12.250][10.227.12.226
   10.227.12.239]{10.227.12.208/255.255.255.240}{10.227.12.240/255.255.2
   55.240}{10.227.12.224/255.255.255.240})
   ```

Syntax of poolRepublish Command

The syntax for the poolRepublish command is:

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

<vrName>

- Name of the VR.
- Value—Text string (value is case sensitive and must match the name in the JUNOSe configuration)
- Guideline—You must enter a value for this property.
- Example—vr-boston

<routerName>

- Name of JUNOSe router on which VR is configured.
- Value—Text string (value is case sensitive and must match the name in the JUNOSe configuration)
- Example—erx1

<ipAddress>

- VR's 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 the 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

- Displays help for this tool.

Troubleshooting the poolRepublish Command

You must specify the correct arguments for the **poolRepublish** command. In addition, the specified router and directory must be available for the command to run successfully.

If no SNMP read community is configured in the directory for the VR and you do not specify this value when you run the **poolRepublish** command, you will see the following error message:

```
Could not perform ip pools update due to No 'snmpReadCommunity' attribute is
provided for virtual router: vr1@bigfoot
```

If you run the **poolRepublish** command again and supply this SNMP read community, the command should run correctly.

Accessing the Router CLI

You can access the CLIs of Juniper Networks routers from Policy Editor and from SDX Admin through a Telnet or SSH connection. This access allows you to display and change the configuration of the router.

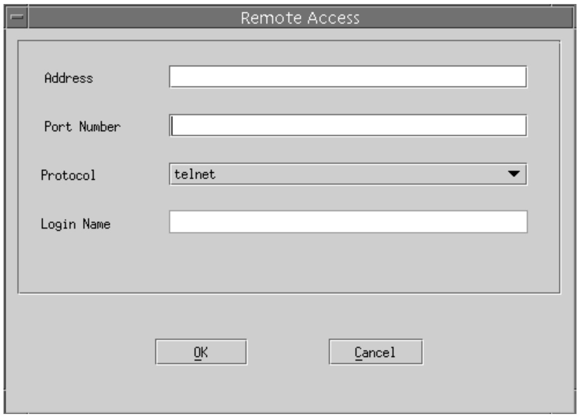
You must have the Telnet or SSH applications installed and available to Policy Editor or SDX Admin. You can open multiple Telnet or SSH sessions.

Using Policy Editor

To access a router from Policy Editor:

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

   The Remote Access dialog box appears.



Protocol: telnet

2. Fill in the Remote Access fields, and click **OK**.

   See *Remote Access Fields* on page 84.

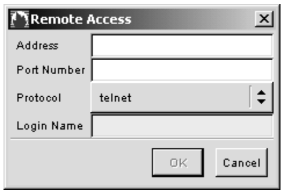
   A Telnet or an SSH window with a CLI prompt appears.

Using SDX Admin

To access a router from SDX Admin:

1. In the navigation pane, expand **o = Network**.
2. Select the router to which you want to connect, and right-click.
3. Select **Manage**.

   The Remote Access dialog box appears.



Protocol telnet

4. Fill in the Remote Access fields, and click **OK**.

   See *Remote Access Fields* on page 84.

   A Telnet or an SSH window with a CLI prompt appears.

Remote Access Fields

In Policy Editor, you can edit the following fields in the Remote Access dialog box, in the Tools > Manage menu.

In SDX Admin, you can edit the following fields in the Remote Access dialog box by right-clicking on the router object, and selecting Manage.

Address

- IP address or hostname of the router.
- Value—IP address
- Default—No value
- Example—192.0.2.1

Port Number

- TCP port over which you want to connect to the router.
- Value—TCP port
- Default—No value
- Example—22

Protocol

- Type of connection
- Value—telnet | ssh
- Default—telnet
- Example—ssh

Login Name

- Login name for SSH connections.
- Value—Text string
- Default—No value
- Guideline—You must enter a value for this property.
- Example—admin

Starting the SRC Client on a JUNOSe Router

JUNOSe routers use an SRC client to interact with the SAE. See *JUNOSe Broadband Access Configuration Guide* for complete information about configuring the SRC client on the JUNOSe router.

To start the SRC client:

1. Access the router CLI.
2. Access Global configuration mode.

   ```
   host1#configure terminal
   ```

3. Switch to the virtual router for which you want to create an SRC client.

   ```
   host1(config)#virtual-router <vrName>
   ```

4. Enable the SRC client.

   To enable COPS-PR mode:

   ```
   host1:<vrName>(config)#sscc enable cops-pr
   ```

   To enable COPS-XDR mode:

   ```
   host1:<vrName>(config)#sscc enable
   ```

5. Set the primary address from the configuration directory.

   ```
   host1:<vrName>(config)#sscc primary address <ipAddress> port 3288
   ```

Stopping the SRC Client on a JUNOSe Router

JUNOSe routers use an SRC client to interact with the SAE. See *JUNOSe Broadband Access Configuration Guide* for complete information about configuring the SRC client on the JUNOSe router.

To stop the SRC client:

1. Access the router CLI.

   See *Accessing the Router CLI* on page 83.

2. Access Global configuration mode.

   ```
   host1#configure terminal
   ```

3. Switch to the virtual router for which you want to stop an SRC client.

   ```
   host1(config)#virtual-router <vrName>
   ```

4. Disable the SRC client.

   ```
   host1:<vrName>(config)#no sscc enable
   ```

Monitoring Interactions Between the SAE and the JUNOSe Router

To monitor the connection between the router and the SAE:

- Use the **show sscc info** command on the JUNOSe router

To display the version number of the SRC client:

- Use the **show sscc version** command on the JUNOSe router.

See the *JUNOSe Command Reference Guides* for details about these commands.

You can also monitor the interactions between the SRC software and the router in the log files for the SAE and in the log files generated by the JUNOSe router. For information about configuring logging for the SAE, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 5, Managing SRC Log Files on a Solaris Platform*. For information about configuring logging on JUNOSe routers, see the *JUNOSe System Event Logging Reference Guide.*

Troubleshooting the SRC Client on JUNOSe Routers

To troubleshoot SRC problems on the router:

1. Look at the log files for the SAE and the log files generated by the SRC client on the JUNOSe router.
   - If the log files indicate a problem with specific interfaces on the router, review the configuration of the associated policies in the SRC software, and fix any errors.
   - If the log files indicate a problem with a specific service or its associated policy rules, review the configuration of the service or policies in the SRC software, and fix any errors.
   - If the log files indicate only that the SRC client is not responding, ensure that the values in the SAE configuration match the values in the SRC client configuration on the router.
2. Restart the SRC client on the JUNOSe router.

   When you restart the SRC client, the SRC client removes all policies that were installed by the SRC software and reports all interfaces again.

NOTE: DHCP addresses that were managed are not reported again, so we recommend that you do not restart the SRC client if you are managing DHCP sessions.

To restart the SRC client in COPS-PR mode, enter the following commands:

```
host1:<vrName>(config)#no sscc enable
host1:<vrName>(config)#sscc enable cops-pr
```

To restart the SRC client in COPS XDR mode, enter the following commands:

```
host1:<vrName>(config)#no sscc enable
host1:<vrName>(config)#sscc enable
```

If restarting the SRC client does not resolve the problem, rebuild the router configuration and restart the client.

Chapter 7

Using JUNOS Routing Platforms in the SRC Network with the SRC CLI

This chapter describes how to use the SRC CLI to set up the SRC software and how to set up JUNOS routing platforms so that the routing platforms can be used the SRC network. It also shows how to monitor the interactions between the SAE and JUNOS routing platforms and troubleshoot SRC problems on JUNOS routing platforms.

You can also use the following to configure JUNOS routers:

- To use the C-Web interface, see *SRC-PE C-Web Interface Configuration Guide, Chapter 18, Using JUNOS Routing Platforms in the SRC Network with the C-Web Interface*.
- To use the Solaris platform, see *Chapter 8, Using JUNOS Routing Platforms in the SRC Network with a Solaris Platform*.

Topics in this chapter include:

- Monitoring Interactions Between the SAE and JUNOS Routing Platforms on page 108
- Troubleshooting Problems with the SRC Software Process on page 109

BEEP Connection Between JUNOS Routing Platforms and the SAE

For information about which JUNOS routing platforms and releases a particular SRC release supports, see the SRC *Release Notes*.

The SAE interacts with a JUNOS software process, referred to as the SRC software process in this documentation, on the JUNOS routing platform. The SAE and the SRC software process communicate using the Blocks Extensible Exchange Protocol (BEEP). You can secure the BEEP connection by using Transport Layer Security (TLS).

When the SRC software process establishes a BEEP session for the SAE, the SAE configures an interface on the JUNOS routing platform. The SAE builds the configuration for an interface using the policies stored in the directory. If the policies are subsequently modified, the SAE builds a new configuration and reconfigures the interface on the JUNOS routing platform.The JUNOS routing platform stores data about interfaces and services that the SAE manages in a configuration group called sdx. You must create this configuration group on the JUNOS routing platform.

Adding JUNOS Routing Platforms and Virtual Routers

On JUNOS routing platforms, the SAE manages interfaces. The SRC software associates a virtual router called default with each JUNOS routing platform. Each JUNOS routing platform in the SRC network and its associated virtual router (VR) called default must appear in the directory. The VRs are not actually configured on the JUNOS routing platform; the VR in the directory provides a way for the SAE to manage the interfaces on the JUNOS routing platform.

There are two ways to add routers:

- Detect operative routers and configured JUNOS VRs in the SRC network and add them to the configuration.
- Add each router and VR individually.

Adding Operative JUNOS Routing Platforms

To add to the directory routers and JUNOS VRs that are currently operative and have an operating SNMP agent:

- In operational mode, enter the following command:

  ```
  request network discovery network network <community community>
  ```

  where:

  - *network*—Address (with or without mask) of the network to discover
  - *community*—Name of the SNMP community to which the devices belong

If you add a router using the discover network feature, the software adds the IP address of the first SNMP agent on the router to respond to the discover request.

Adding Routers Individually

Use the following configuration statements to add a router device:

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

To add a router device:

1. From configuration mode, access the configuration statements that configure network devices. This procedure uses junos_boston as the name of the router.

   ```
   user@host# edit shared network device junos_boston
   ```

2. (Optional) Add a description for the router.

   ```
   [edit shared network device junos_boston]
   user@host# set description description
   ```

3. (Optional) Add the IP address of the router.

   ```
   [edit shared network device junos_boston]
   user@host# set management-address management-address
   ```

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

   ```
   [edit shared network device junos_boston]
   user@host# set device-type junos
   ```

5. (Optional) Verify your configuration.

   ```
   [edit shared network device junos_boston]
   user@host# show
   description "This is a core-facing JUNOS router.";
   management-address 10.117.8.32;
   device-type junos;
   interface-classifier {
     rule rule-0 {
       script #;
     }
   }
   ```

Adding Virtual Routers Individually

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...];
    tracking-plug-in [tracking-plug-in...];
}
```

To add a virtual router:

1. From configuration mode, access the configuration statements for virtual routers. This procedure uses junos_Boston as the name of the router. For JUNOS routing platforms, use the name default for the virtual router.

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

   To specify the active SAE and the redundant SAE, enter an exclamation point (!) after the hostname or IP address of the connected SAE. For example:

   ```
   [edit shared network device junos_boston virtual-router vr1]
   user@host# set sae-connection [sae1! sae2!]
   ```

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

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

5. (Optional) Specify service scopes assigned to this virtual router. The scopes are available for subscribers connected to this virtual router for selecting customized versions of services.

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

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

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

7. (Optional) Verify your configuration.

```
[edit shared network device junos_boston virtual-router default]
user@host# show
sae-connection 192.168.80.1;
snmp-read-community ********;
snmp-write-community ********;
scope POP-Cambridge;
tracking-plug-in flexRadius;
```

Related Topics

- For information about service scopes, see *SRC-PE Services and Policies Guide, Chapter 1, Managing Services with the SRC CLI*
- For information about tracking plug-ins, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 11, Configuring Accounting and Authentication Plug-Ins with the SRC CLI*

Configuring the SAE to Manage JUNOS Routing Platforms

A JUNOS routing platform interacts with the SAE by using a JUNOS software process called sdxd. When the sdxd process establishes a TCP/IP connection to the SAE, the SAE begins to manage the router. The JUNOS router driver configuration defines parameters related to the interactions between the SAE and the sdxd process.

Use the following configuration statements to configure the JUNOS router driver:

shared sae configuration driver junos {

```
    beep-server-port beep-server-port;
    tls-beep-server-port tls-beep-server-port;
    connection-attempts connection-attempts;
    keepalive-interval keepalive-interval;
    message-timeout message-timeout;
    batch-size batch-size;
    transaction-batch-time transaction-batch-time;
    sdx-group-name sdx-group-name;
    sdx-session-group-name sdx-session-group-name;
    send-commit-check send-commit-check;
}
```

To configure the JUNOS router driver:

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

   ```
   user@host# edit shared sae group west-region configuration driver junos
   ```

2. Specify the TCP port number that is used to communicate with the sdxd process on JUNOS routing platforms. This port number must match the port number configured in the sdxd process on the router.

   If you set this value to zero and the TLS BEEP server port is set, the SAE accepts only TLS connections.

   ```
   [edit shared sae group west-region configuration driver junos]
   user@host# set beep-server-port beep-server-port
   ```

3. Specify the TLS port number that is used for TLS connections to the JUNOS routing platform.

   If you set this value to zero, the SAE does not accept TLS connections.

   ```
   [edit shared sae group west-region configuration driver junos]
   user@host# set tls-beep-server-port tls-beep-server-port
   ```

4. Specify the number of outstanding connection attempts before new connection attempts are dropped.

   ```
   [edit shared sae group west-region configuration driver junos]
   user@host# set connection-attempts connection-attempts
   ```

5. Specify the interval between keepalive messages sent from the router.

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

6. Specify the amount of time that the router driver waits for a response from the sdxd process.

   Under a high load the router may not be able to respond fast enough to requests. Change this value only if a high number of timeout events appear in the error log.

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

7. Specify the minimum number of service configuration transactions that are committed at the same time

```
[edit shared sae group west-region configuration driver junos]
user@host# set batch-size batch-size
```

8. Specify the maximum time to collect configuration transactions in a batch.

```
[edit shared sae group west-region configuration driver junos]
user@host# set transaction-batch-time transaction-batch-time
```

9. Specify the name of a session group on the JUNOS routing platform in which provisioning objects are stored.

```
[edit shared sae group west-region configuration driver junos]
user@host# set sdx-session-group-name sdx-session-group-name
```

10. Enable or disable commit check. If enabled, a more detailed error message is logged if a batch fails, which lets you verify individual transactions in a batch.

```
[edit shared sae group west-region configuration driver junos]
user@host# set send-commit-check send-commit-check
```

11. (Optional) Verify your configuration.

```
[edit shared sae group west-region configuration driver junos]
user@host# show
beep-server-port 3333;
tls-beep-server-port 0;
connection-attempts 50;
keepalive-interval 45;
message-timeout 30000;
batch-size 10;
transaction-batch-time 2000;
sdx-group-name sdx;
sdx-session-group-name sdx-sessions;
send-commit-check true;
```

Related Topics

- For information about setting up SAE groups, see *SRC-PE Getting Started Guide, Chapter 21, Setting Up an SAE with the SRC CLI.*

Configuring Secure Connections Between the SAE and JUNOS Routing Platforms

You can use TLS to protect communication between the SAE and JUNOS routing platforms.

To complete the handshaking protocol for the TLS connection, the client (JUNOS routing platform) and the server (SAE) must exchange and verify certificates. You need to create a client certificate and a server certificate. Both certificates must be signed by a certificate authority (CA). JUNOS software supports VeriSign, Inc. (http://www.verisign.com). You must then install both certificates on the SAE and on the JUNOS routing platform.

You can use SRC CLI commands to manage certificates manually, or through the Simple Certificate Enrollment Protocol (SCEP).

Certificates are in the format defined in the X.509 standard for public key infrastructure. The certificate requests are in the Public Key Cryptology Standard (PKCS) #10 format.

Tasks to set up the SAE and the JUNOS routing platform to use TLS are:

1. Manually Obtaining Digital Certificates on page 96

   Or

2. Obtaining Digital Certificates through SCEP on page 98
3. Installing the Server Certificate on the Routing Platform on page 99
4. Creating a Client Certificate for the Router on page 100
5. Installing the Client Certificate on the Router on page 100
6. Configuring the SAE to Use TLS on page 100
7. Configuring TLS on the SAE on page 100

Manually Obtaining Digital Certificates

You can manually add digital certificates, or you can use SCEP to help manage how you obtain certificates. See *Obtaining Digital Certificates through SCEP* on page 98.

To manually add a signed certificate:

1. Create a certificate signing request.

   user@host> **request security generate-certificate-request subject** *subject* **password** *password*

   where:

   - subject is the distinguished name of the SRC host; for example cn=src1,ou=pop,o=Juniper,l=kanata, st=Ontario,c=Canada.

- password is the password received from the certificate authority for the specified subject.

By default, this request creates the file /tmp/certreq.csr and encodes the file by using Privacy-Enhanced Mail (pem) encoding.

2. Copy the file generated in Step 1 to another system, and submit the certificate signing request file generated in Step 1 to VeriSign, Inc. (http://www.verisign.com) for signing.

   You can transfer the file through FTP by using the file copy command.

   user@host> **file copy** *source_file* **ftp://***username***@***server*[**:***port*]**/***destination_file*

   VeriSign authenticates you and returns a certificate, signed by them, that authenticates your public key.

3. When you receive the signed certificate, copy the file back to the SRC system to the */tmp* directory.

   You can transfer the file through FTP, as shown in Step 2.

4. Add the certificate to the SRC configuration.

   user@host> **request security import-certificate file-name** *file-name* **identifier** *identifier*

   where:

   - file-name is the name of the certificate file in the */tmp* folder. The file must be in one of the following formats, which is indicated by the following extensions:
     - CER—Windows extension
     - PEM—Privacy-Enhanced Mail encoding
     - DER—Binary encoding
     - BER—Binary encoding
   - identifier is the name of the certificate.

   For example, to import the file src.cer that is identified as web:

   user@host> **request security import-certificate file-name src.cer identifier web**

5. Verify that the certificate is part of the SRC configuration.

```
user@host> show security certificate
web subject:CN=host
```

   If there are no certificates on the system, the CLI displays the following message:

```
No entity certificates in key store
```

Obtaining Digital Certificates through SCEP

You can use SCEP to help manage how you obtain digital certificates, or you can manually add certificates. See *Manually Obtaining Digital Certificates* on page 96.

Before you can obtain certificates for your use, you must get the CA's certificate and install it in the local store of trusted certificates.

To add a signed certificate that you obtain through SCEP:

1. Request your CA's certificate through SCEP.

   ```
   user@host> request security get-ca-certificate url url ca_identifier ca_identifier
   ```

   where:

   - url is the URL of the certificate authority (which is the SCEP server).
   - ca-identifier is the identifier that designates the authority.

   For example, to request a certificate from the CA authority SrcCA at a specified URL on the server security_server:

   ```
   user@host> request security get-ca-certificate url
   http://security_server:8080/ejbca/publicweb/apply/scep/pkiclient.exe
   ca-identifier SrcCA

   Version: 3
   Serial Number: 5721058705923989279
   Signature Algorithm: SHA1withRSA
   Issuer: CN=SrcCA
   Valid From: Wed Sep 06 17:00:55 EDT 2006
   Valid Until: Sat Sep 03 17:10:55 EDT 2016
   Subject: CN=SrcCA
   Public key: RSA
   Thumbprint Algorithm: SHA1
   Thumbprint: 3c 57 a9 77 af 83 3 e9 c7 1e ee e2 4a e8 ff f3 89 f4 11 a9
   Do you want to add the above certificate as a trusted CA [yes,no] ? (no) y
   ```

2. Request that the certificate authority automatically sign the certificate request.

   ```
   user@host> request security enroll subject subject password password
   ```

   where:

   - subject is the distinguished name of the SRC host; for example cn=myhost.
   - password is the password received from the certificate authority.

   For example, to request a certificate from the CA authority SrcCA at a specified URL on the server security_server:

   ```
   user@host> request security enroll url
   http://security_server:8080/ejbca/publicweb/apply/scep/pkiclient.exe
   identifier web ca-identifier SrcCA subject cn=myhost password mypassword
   ```

```
Received certificate:
Version: 3
Serial Number: 6822890691617224432
Signature Algorithm: SHA1withRSA
Issuer: CN=SrcCA
Valid From: Tue Sep 19 16:33:11 EDT 2006
Valid Until: Thu Sep 18 16:43:11 EDT 2008
Subject: CN=myhost
Public key: RSA
Do you want to install the above certificate [yes,no] ? (no) y
```

3. Verify that the certificate is part of the SRC configuration.

```
user@host> show security certificate
web subject:CN=myhost
```

If there are no certificates on the system, the CLI displays the following message:

```
No entity certificates in key store
```

Installing the Server Certificate on the Routing Platform

The TLS client (JUNOS routing platform) needs a copy of the certificate that was used to sign the SAE certificate so that it can verify the SAE certificate. To install the SAE certificate on the JUNOS routing platform:

1. Include the following statements at the [edit security certificates certificate-authority] hierarchy level.

```
[edit security certificates certificate-authority]
security{
    certificates{
        certificate-authority SAECert{
          file /var/db/certs/cert.pem;
        }
    }
}
```

2. Include the following statements at the [system services service-deployment] hierarchy level.

```
system{
    services{
      service-deployment{
          servers {
            server-address port port-number{
              security-options {
                tls;
              }
            }
          }
      }
    }
}
```

Creating a Client Certificate for the Router

For information about how to obtain a certificate for the router from a certificate authority, see *Obtaining a Certificate from a Certificate Authority* in the *JUNOS System Basics Configuration Guide.*

Installing the Client Certificate on the Router

To install the client (router) certificate on the JUNOS routing platform:

1. Include the following statements at the [edit security certificates certificate-authority] hierarchy level.

```
[edit security certificates certificate-authority]
security{
    certificates{
            local clientCERT { .... } ;
    }
}
```

2. Include the following statements at the [system services service-deployment] hierarchy level.

```
system{
    services{
            service-deployment{
                local-certificate clientCert;
            }
    }
}
```

Configuring the SAE to Use TLS

To configure the SAE to accept TLS connections, enter a port number with the **set beep-server-port** command in the JUNOS router driver configuration.

See *Configuring the SAE to Manage JUNOS Routing Platforms* on page 93.

Configuring TLS on the SAE

Use the following configuration statements to configure TLS on the SAE:

```
shared sae configuration driver junos security {
    need-client-authentication;
    certificate-identifier private-key;
}
```

To configure TLS on the SAE:

1. From configuration mode, access the configuration statement that configures security for the JUNOS TLS connection. In this sample procedure, the JUNOS driver is configured in the west-region group.

```
user@host# edit shared sae group west-region configuration driver junos security
```

2. (Optional) Specify whether or not the SAE requests a client certificate from the router when a connection to the router is established.

   ```
   [edit shared sae group west-region configuration driver junos security]
   user@host# set need-client-authentication
   ```

3. Specify the name of certificate to be used for TLS communications.

   ```
   [edit shared sae group west-region configuration driver junos security]
   user@host# set certificate-identifier private-key
   ```

4. (Optional) Verify your TLS configuration.

   ```
   [edit shared sae group west-region configuration driver junos security]
   user@host# show
   need-client-authentication;
   certificate-identifier privatekey;
   ```

Checking Changes to the JUNOS Configuration

The SAE can check the configuration of a JUNOS routing platform under its control to detect whether the configuration has changed by a means other than through the SAE. If the SAE finds a disparity between the router and the SAE configurations, it can take several actions. The SAE checks the configuration installed on the router against the state of the SAE session layer (subscriber, service, and interface sessions). While the check is occurring, the SAE does not handle jobs from the router, and all provisioning activity is blocked, including event notifications.

The SAE can take the following actions if it finds a disparity between the router and SAE configurations:

- The SAE takes the state of the session layer on the router to be correct and updates its local state to be consistent with the router. The SAE then sends stop events for all sessions where the corresponding provisioning in the router has been removed.

- The SAE takes its local state to be the correct state and updates the router to be consistent with its local state.

- The SAE does not solve the state discrepancy. It reports disparities through the SAE device driver event trap called routerConfOutOfSynch and through the info log.

Note that it is not possible to check the consistency of individual objects that the SAE provisions Therefore, modifications to a provisioning object while the SAE is disconnected from the router cannot be detected.

Setting Up Periodic Configuration Checking

Use the following configuration statements to configure the SAE to periodically check the configuration of the JUNOS routing platform:

```
shared sae configuration driver junos configuration-checking
    configuration-checking-schedule configuration-checking-schedule;
    configuration-checking-action (enforce | synchronize | detect);
}
```

To configure the SAE to periodically check the configuration of the JUNOS routing platform:

1. From configuration mode, access the configuration statement that configures the configuration checking feature.

   ```
   user@host# edit shared sae configuration driver junos configuration-checking
   ```

2. Specify when the SAE checks the router configuration.

   ```
   [edit shared sae configuration driver junos configuration-checking]
   user@host# set configuration-checking-schedule configuration-checking-schedule
   ```

3. Specify the action that the SAE takes when it detects disparities between the configuration of the SAE and the configuration on the router.

   ```
   [edit shared sae configuration driver junos configuration-checking]
   user@host# set configuration-checking-action enforce | synchronize | detect
   ```

4. (Optional) From operational mode, verify your configuration checking configuration.

   ```
   [edit shared sae configuration driver junos configuration-checking]
   user@host# show
   configuration-checking-schedule "0 0 * * * * *";
   configuration-checking-action synchronize;
   ```

Using SNMP to Retrieve Information from JUNOS Routing Platforms

You can use SNMP to retrieve information from the router. For example, if you create a router initialization script that uses SNMP, you need to specify the SNMP communities that are on the router.

We recommend that you specify SNMP communities for each virtual router. (See *Adding Virtual Routers Individually* on page 92.) You can also configure global default SNMP communities.

Configuring Global SNMP Communities in the SRC Software

You can configure global default SNMP communities that are used if a VR does not exist on the router or if the community strings have not been configured for the VR.

Use the following configuration statements to configure global default SNMP communities:

```
shared sae configuration driver snmp {
    read-only-community-string read-only-community-string;
    read-write-community-string read-write-community-string;
}
```

To configure global default SNMP communities:

1. From configuration mode, access the configuration statements that configure default SNMP communities.

   ```
   user@host# edit shared sae configuration driver snmp
   ```

2. Configure the default SNMP community string used for read access to the router.

   ```
   [edit shared sae configuration driver snmp]
   user@host# set read-only-community-string read-only-community-string
   ```

3. Configure the default SNMP community string used for write access to the router.

   ```
   [edit shared sae configuration driver snmp]
   user@host# set read-write-community-string read-write-community-string
   ```

4. (Optional) Verify your configuration.

   ```
   [edit shared sae configuration driver snmp]
   user@host# show
   read-only-community-string ********;
   read-write-community-string ********;
   ```

Developing Router Initialization Scripts

When the SAE establishes a connection with a router, it can run a router initialization script to customize the setup of the connection. Router initialization scripts are run when the connection between a router and the SAE is established and again when the connection is dropped.

We provide the iorPublisher script in the */opt/UMC/sae/lib* folder. The iorPublisher script publishes the IOR of the SAE into an internal part of the shared configuration so that a NIC can associate a router with an SAE.

Interface Object Fields

Router initialization scripts are written in the Python programming language (www.python.org) and executed in the Jython environment (www.jython.org).

Router initialization scripts interact with the SAE through an interface object called Ssp. The SAE exports a number of fields through the interface object to the script and expects the script to provide the entry point to the SAE.

Table 8 describes the fields that the SAE exports.

Table 8: Exported Fields

Ssp Attribute	Description
Ssp.properties	System properties object (class: java.util.Properties)—The properties should be treated as read-only by the script.
Ssp.errorLog	Error logger—Use the SsperrorLog.printIn (message) to send error messages to the log.
Ssp.infoLog	Info logger—Use the Ssp.infoLog.printIn (message) to send informational messages to the log.
Ssp.debugLog	Debug logger—Use the Ssp.debugLog.printIn (message) to send debug messages to the log.

The router initialization script must set the field Ssp.routerInit to a factory function that instantiates a router initialization object:

- <VRName>—Name of the virtual router in which the COPS client has been configured, in the format: virtualRouterName@RouterName
- <virtualIp>—Virtual IP address of the SAE (string, dotted decimal; for example: 192.168.254.1)
- <realIp>—Real IP address of the SAE (string, dotted decimal; for example, 192.168.1.20)
- <VRIp>—IP address of the virtual router (string, dotted decimal)
- <transportVR>—Name of the virtual router used for routing the COPS connection, or None, if the COPS client is directly connected

The factory function must implement the following interface:

```
Ssp.routerInit(VRName,
               virtualIp,
               realIp,
               VRIp,
               transportVR)
```

The factory function returns an interface object that is used to set up and tear down a connection for a given COPS server. A common case of a factory function is the constructor of a class.

The factory function is called directly after a COPS server connection is established. In case of problems, an exception should be raised that leads to the termination of the COPS connection.

Required Methods

Instances of the interface object must implement the following methods:

- *setup()*—Is called when the COPS server connection is established and is operational. In case of problems, an exception should be raised that leads to the termination of the COPS connection.
- *shutdown()*—Is called when the COPS server connection is terminated to the virtual router. This method should not raise any exceptions in case of problems.

Example: Router Initialization Script

The following script defines a router initialization class named *SillyRouterInit*. The interface class does not implement any useful functionality; it just writes messages to the infoLog when the router connection is created or terminated.

```
class SillyRouterInit:
    def __init__(self, vrName, virtualIp, realIp, vrIp, transportVr):
        """ initialize router initialization object """
        self.vrName = vrName
        Ssp.infoLog.printin("SillyRouterInit created")

    def setup(self):
        """ initialize connection to router """
        Ssp.infoLog.printin("Setup connection to VR %(vrName)s" %
                    vars(self))

    def shutdown(self):
        """ shutdown connection to router """
        Ssp.infoLog.printin("Shutdown connection to VR %(vrName)s" %
                    vars(self))

#
# publish interface object to Ssp core
#
Ssp.routerInit = SillyRouterInit
```

Specifying Router Initialization Scripts on the SAE

Use the following configuration statements to specify router initialization scripts for JUNOS routing platforms:

```
shared sae configuration driver scripts {
    extension-path extension-path;
    general general;
    junos junos;
}
```

To configure router initialization scripts for JUNOS routing platforms:

1. From configuration mode, access the configuration statements that configure router initialization scripts. In this sample procedure, the scripts are configured in the west-region group.

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

2. Specify the router initialization script for JUNOS routing platforms.

   ```
   [edit shared sae group west-region configuration driver scripts]
   user@host# set junos junos
   ```

3. Configure a router initialization script that can be used for all types of routers that the SRC software supports.

   ```
   [edit shared sae group west-region configuration driver scripts]
   user@host# set general general
   ```

4. Configure a path to router initialization scripts that are not in the default location, */opt/UMC/sae/lib*.

   ```
   [edit shared sae group west-region configuration driver scripts]
   user@host# set extension-path extension-path
   ```

5. (Optional) From operational mode, verify your router initialization script configuration.

   ```
   [edit shared sae group west-region configuration driver scripts]
   user@host# show
   extension-path ;
   junos iorPublisher;
   ```

Accessing the Router CLI

You can access the CLIs of Juniper Networks routers through a Telnet or secure shell connection.

- To open a Telnet session to a router, use the telnet operational mode command. For example:

  ```
  user@host> telnet 10.10.10.3
  ```

- To open a secure shell connection, use the ssh operational command. For example:

  ```
  user@host> ssh host 10.10.10.3
  ```

Configuring JUNOS Routing Platforms to Interact with the SAE

To configure the JUNOS routing platform to interact with the SAE:

1. Include the following statements at the [edit system services service-deployment] hierarchy level.

```
[edit system services service-deployment]
servers server-address {
port port-number;
}
source-address source-address;
```

2. Use the following guidelines for the variables in these statements.

server-address

- Specifies the IP address of the host on which you install the SAE.
- Value—IP address
- Guidelines—Be sure this setting matches the corresponding value in the SAE configuration.
- Default—None
- Example—192.0.2.2

port-number

- Specifies the port number for the SAE.
- Value—TCP port number
- Guidelines—Be sure this setting matches the corresponding value in the SAE configuration.
- Default—3333
- Example—3333

source-address

- Specifies the IP address of the source that sends traffic to the SAE.
- Value—IP address
- Guidelines—This setting is optional.
- Default—None
- Example—192.0.2.2

Configuring the JUNOS Routing Platform to Apply Changes It Receives from the SAE

To configure the JUNOS routing platform to receive configuration statements from the SAE and apply those statements to the configuration:

1. Create a configuration group called sdx that contains the configuration statements that the SAE sends to the JUNOS routing platform. To do so, include the groups statement at the [edit] level, and specify the name sdx.

   ```
   [edit]
   groups {
       sdx;
   }
   ```

2. Configure the JUNOS routing platform to apply these statements to the configuration. To do so, include the apply-groups statement at the [edit] level.

   ```
   [edit]
   set apply-groups sdx;
   ```

Disabling Interactions Between the SAE and JUNOS Routing Platforms

To disable the SRC software process, enter the following command:

```
root@ui1#set system processes service-deployment disable
root@ui1#commit
```

When you disable the SRC software process, it is still available on the JUNOS routing platform.

To reenable the SRC software process, enter the following command:

```
root@ui1#delete system processes service-deployment disable
root@ui1#commit
```

The SRC software process attempts to reconnect the JUNOS routing platform to the SAE.

Monitoring Interactions Between the SAE and JUNOS Routing Platforms

Use the following command on JUNOS routing platforms to monitor the connection between the JUNOS routing platform and the SAE.

```
root@ui1> show system services service-deployment
Connected to 172.17.20.151 port 3333 since 2004-02-06 14:50:31 PST
Keepalive settings: Interval 15 seconds
Keepalives sent: 100, Last sent: 6 seconds ago
Notifications sent: 0
Last update from peer: 00:00:06 ago
```

You can also monitor the interactions between the SRC software and JUNOS routing platforms in the log files for the SAE and in the log files generated by the SRC software process on the JUNOS routing platform.

- For information about configuring logging for the SAE, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 3, Configuring Logging for SRC Components with the CLI*.
- For information about configuring logging on JUNOS routing platforms, see *JUNOS System Basics Configuration Guide*.

Troubleshooting Problems with the SRC Software Process

To troubleshoot SRC problems on the JUNOS routing platform, review the log files for the SAE and the log files generated by the SRC software process on the router. If the log files indicate that the SRC software process on the JUNOS routing platform is not responding:

1. Look at the status of the process on the JUNOS routing platform.

```
root@ui1>show system services service-deployment
Connected to 172.17.20.151 port 3333 since 2004-02-06 14:50:31 PST
Keepalive settings: Interval 15 seconds
Keepalives sent: 100, Last sent: 6 seconds ago
Notifications sent: 0
Last update from peer: 00:00:06 ago
```

2. If you see the message “error: the service-deployment subsystem is not running,” reenable the SRC software process. See *Disabling Interactions Between the SAE and JUNOS Routing Platforms* on page 108.
3. If the process is already enabled, review the configurations of the router and the SAE in the directory, and fix any problems.
4. Restart the SRC software process on the router.

```
root@ui1>restart service-deployment
```

   The SAE synchronizes with the SRC software process and deletes unnecessary data from the router.

Deleting All SRC Data on JUNOS Routing Platforms

If deleting parts of the SRC data on a JUNOS routing platform fails to solve problems, delete all the SRC data and restart the SRC software process. To do so:

1. Delete all SRC interfaces and services.

```
delete groups sdx
root@ui1#commit
```

2. If you are running SDX software Releases 5.0 through 6.1, you should also delete interface sessions. (After Release 6.2, session data is no longer stored on the router, it is stored on the SAE host using the session store feature.)

```
delete groups sdx-sessions
root@ui1#commit
```

3. Restart the SRC software process on the router.

   root@ui1>**restart service-deployment**

Viewing the State of JUNOS Device Drivers with the SRC CLI

To display the state of JUNOS drivers, use the following operational mode command:

show sae drivers <device-name *device-name*> < (brief) > <maximum-results *maximum-results*>

For example:

```
user@host> show sae drivers device-name default@jrouter
JUNOS Driver
Device name                           default@jrouter
Device type                           junos
Device IP                             /10.10.6.113:1879
Local IP                              10.10.6.113
TransportRouter
Device version                        8.2R1.7
Start time                            Thu Mar 08 21:00:50 UTC 2007
Number of notifications               0
Number of processed added             0
Number of processed changed           0
Number of processed deleted           0
Number of provisioning attempt        0
Number of provisioning attempt failed 0
Device type                           JunosRouterDriver
Job queue size                        0
Number of SAP                         3
Number of PAP                         0
Start time                            Thu Mar 08 21:00:55 UTC 2007
End time                              Thu Mar 08 21:00:55 UTC 2007

  Transaction Manager
  Transaction queue size 0
  Router name            default@troll
```

Viewing Statistics for Specific JUNOS Device Drivers with the SRC CLI

To display statistics for a specific JUNOS device driver, use the following operational mode command:

show sae statistics device <name *name*> < (brief) >

For example:

```
user@host> show sae statistics device name default@jrouter
SNMP Statistics
Add notification handle time      7
Change notification handle time   0
Client ID                         default@troll
Delete notification handle time   0
Failover IP                       0.0.0.0
Failover port                     0
Handle message time               40
Job queue age                     0
Job queue time                    0
```

```
Number message send              3
Number of added jobs             0
Number of add notifications      0
Number of change notifications   0
Number of delele notifications   0
Number of managed interfaces     3
Number of message errors         0
Number of message timeouts       0
Number of removed jobs           0
Number of user session established 0
Number of user session removed   0
Router type                      JUNOS
Up time                          7036120
Using failover server            false
```

Viewing Statistics for All JUNOS Device Drivers with the SRC CLI

To display SNMP statistics for all JUNOS device drivers, use the following operational mode command:

```
show sae statistics device common junos
```

For example:

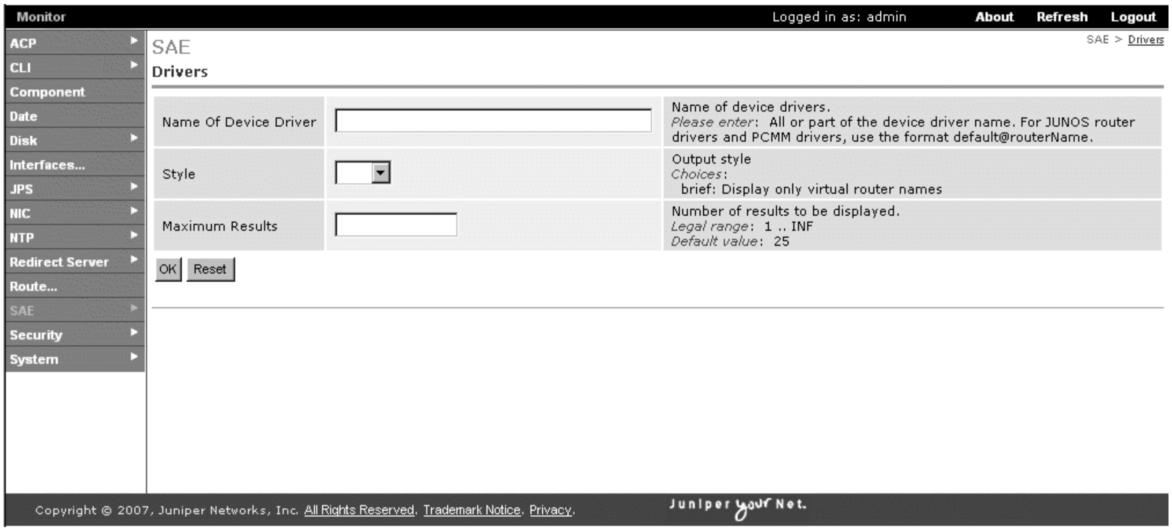
```
user@host> show sae statistics device common junos
SNMP Statistics
Driver type                  JUNOS
Number of close requests     0
Number of connections accepted 0
Number of current connections  0
Number of open requests      0
Server address               0.0.0.0
Server port                  3288
Time since last redirect     0
```

Viewing the State of JUNOS Device Drivers with the C-Web Interface

If the log files indicate a problem with a specific driver, review the configuration of the associated JUNOS router driver with C-Web.

1. Select **SAE** from the side pane, and click **Drivers**.

   The Drivers pane appears.



SAE

Drivers

Name Of Device Driver		Name of device drivers. *Please enter*: All or part of the device driver name. For JUNOS router drivers and PCMM drivers, use the format default@routerName.
Style		Output style *Choices*: brief: Display only virtual router names
Maximum Results		Number of results to be displayed. *Legal range*: 1 .. INF *Default value*: 25

2. In the Name of Device Driver box, enter a full or partial device driver name for which you want to display information, or leave the box blank to display all devices. Use the format:

   default@<router name>

3. Select an output style from the Style list.

4. In the Maximum Results box, enter the maximum number of results that you want to receive.

5. Click **OK**.

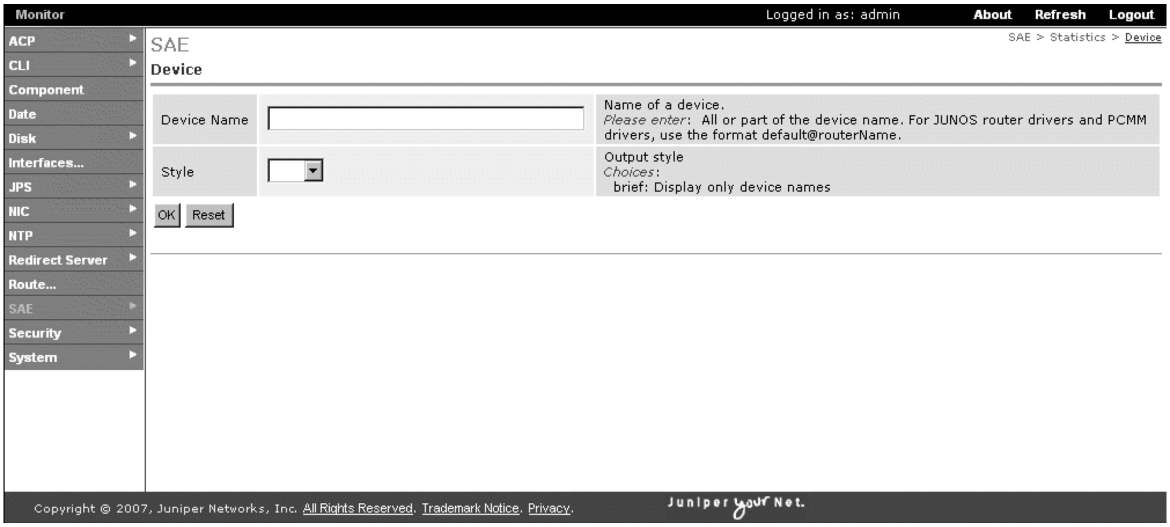
   The Drivers pane displays information about the JUNOS device driver.

Viewing Statistics for Specific JUNOS Device Drivers with the C-Web Interface

To view SNMP statistics about devices:

1. Select **SAE** from the side pane, click **Statistics**, and then click **Device**.

   The Device pane appears.



Device Name		Name of a device. *Please enter*: All or part of the device name. For JUNOS router drivers and PCMM drivers, use the format default@routerName.
Style		Output style *Choices*: brief: Display only device names

2. In the Device Name box, enter a full or partial device name for which you want to display information, or leave the box blank to display all devices.

3. Select an output style from the Style list.

4. Click **OK**.

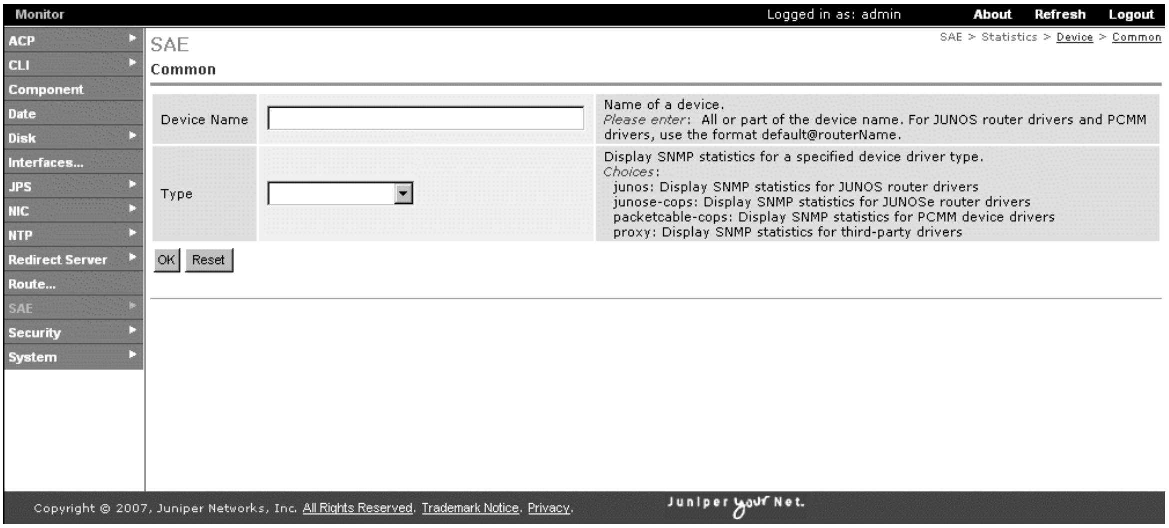
   The Device pane displays statistics for all devices.

Viewing Statistics for All JUNOS Device Drivers with the C-Web Interface

To view SNMP statistics about specific devices:

1. Select **SAE** from the side pane, click **Statistics**, click **Device**, and then click **Common**.

   The Common pane appears.



SAE

Common

Device Name		Name of a device. *Please enter*: All or part of the device name. For JUNOS router drivers and PCMM drivers, use the format default@routerName.
Type		Display SNMP statistics for a specified device driver type. *Choices*: junos: Display SNMP statistics for JUNOS router drivers junose-cops: Display SNMP statistics for JUNOSe router drivers packetcable-cops: Display SNMP statistics for PCMM device drivers proxy: Display SNMP statistics for third-party drivers

2. In the Device Name box, enter a full or partial device name for which you want to display information, or leave the box blank to display all devices.

3. Select the junos from the Type list:

4. Click **OK**.

   The Common pane displays statistics for the specified device.

Chapter 8

Using JUNOS Routing Platforms in the SRC Network with a Solaris Platform

This chapter describes how to set up the SRC software on a Solaris platform with the SRC configuration applications that run only on Solaris platforms. It also shows how to set up JUNOS routing platforms so that the routing platforms can be used the SRC network. It includes information about how to monitor the interactions between the SAE and JUNOS routing platforms and how to troubleshoot SRC problems on JUNOS routing platforms.

You can also use the following to configure JUNOS routers:

- To use the SRC CLI, see *Chapter 7, Using JUNOS Routing Platforms in the SRC Network with the SRC CLI*.
- To use the C-Web interface, see *SRC-PE C-Web Interface Configuration Guide, Chapter 18, Using JUNOS Routing Platforms in the SRC Network with the C-Web Interface*.

Topics in this chapter include:

- Disabling Interactions Between the SAE and JUNOS Routing Platforms on page 135
- Monitoring Interactions Between the SAE and JUNOS Routing Platforms on page 135
- Troubleshooting SRC Problems on JUNOS Routing Platforms on page 136

BEEP Connection Between JUNOS Routing Platforms and the SAE

For information about which JUNOS routing platforms and releases a particular SRC release supports, see the SRC *Release Notes*.

The SAE interacts with a JUNOS software process, referred to as the SRC software process in this documentation, on the JUNOS routing platform. The SAE and the SRC software process communicate using the Blocks Extensible Exchange Protocol (BEEP). You can secure the BEEP connection by using Transport Layer Security (TLS).

When the SRC software process establishes a BEEP session for the SAE, the SAE configures an interface on the JUNOS routing platform. The SAE builds the configuration for an interface using the policies stored in the directory. If the policies are subsequently modified, the SAE builds a new configuration and reconfigures the interface on the JUNOS routing platform.The JUNOS routing platform stores data about interfaces and services that the SAE manages in a configuration group called sdx. You must create this configuration group on the JUNOS routing platform.

Adding JUNOS Routing Platforms and Virtual Routers

On JUNOS routing platforms, the SAE manages interfaces. The SRC software associates a virtual router called default with each JUNOS routing platform. Each JUNOS routing platform in the SRC network and its associated virtual router (VR) called default must appear in the directory. The VRs are not actually configured on the JUNOS routing platform; the VR in the directory provides a way for the SAE to manage the interfaces on the JUNOS routing platform.

There are two ways to add routers to the directory:

- Use SDX Admin to detect operative routers in the SRC network and add them to the directory. This operation creates a VR called default in the directory for each detected JUNOS routing platform.
- Add each router and VR individually. You need to add routers and VRs individually if you use an LDAP client other than SDX Admin or if you want to add inoperative routers.

NOTE: You must define connected SAEs for each router in the virtual router object of the directory. This step is required for the SAE to work with the router. See *Specifying the SAEs That Can Manage the Router* on page 123.

Adding Operative JUNOS Routing Platforms

To add routers that are currently operative and have an operating SNMP agent:

1. In the SDX Admin navigation pane, select **o = Network**, and right-click.
2. Select **Discover Network**.

   The Discover Network dialog box appears.
3. Enter the IP address, the prefix of the network, and the SNMP community string.
4. Click **OK**.

   For each JUNOS routing platform, the software creates one VR called default. You can modify the configuration of these objects. For information about configuring these objects, see *Adding Routers Individually* on page 117 and *Adding Virtual Routers Individually* on page 119.

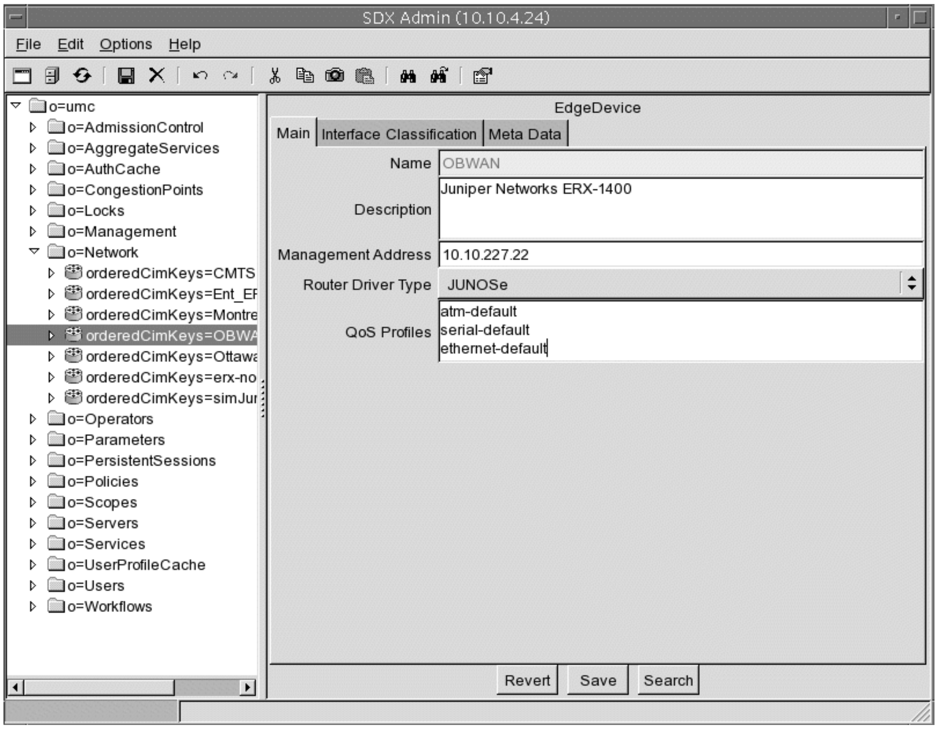
Adding Routers Individually

To add a single router with SDX Admin:

1. In the navigation pane, right-click the Network folder, and select **New > EdgeDevice**.

   The New EdgeDevice dialog box appears.
2. Enter the name of the router exactly as it is configured in the JUNOS software, and click **OK**.

   The new device appears in the navigation pane, and the Main tab of the EdgeDevice pane appears.



Name: OBWAN

Description: Juniper Networks ERX-1400

Management Address: 10.10.227.22

Router Driver Type: JUNOSe

QoS Profiles:
atm-default
serial-default
ethernet-default

3. Edit or accept the default values for the router fields.

   See *Router Fields* on page 118.

4. Click **Save**.

Router Fields

In SDX Admin, you can modify the following fields in the content pane for a router (*orderedCimKeys = <EdgeDeviceName> , o = network, o = umc)*.

Description

- Information about this device; keywords that the SRC find utility uses.
- Value—Text string
- Example—ERX-1400 router located in Ottawa

Management Address

- IP address of the router or CMTS device. If you add a router using the discover network feature, the software automatically adds the IP address of the first SNMP agent on the router to respond to the discover request.
- Value—IP address
- Example—192.0.1.1

Router Driver Type

- Type of device that this directory object will be used to manage.
- Value
  - JUNOSe—JUNOSe router
  - JUNOS—JUNOS routing platform
  - PCMM—CMTS device
- Default—No value

QoS Profiles

- For JUNOSe routers, specifies quality of service (QoS) profiles that are configured on the router.
- Value—List of QoS profiles on separate lines
- Guideline—This field applies to JUNOSe routers only
- Example—atm-default

Adding Virtual Routers Individually

To add a VR with SDX Admin:

1. In the navigation pane, right-click the device to which you want to add the VR, and select **New > VirtualRouter**.

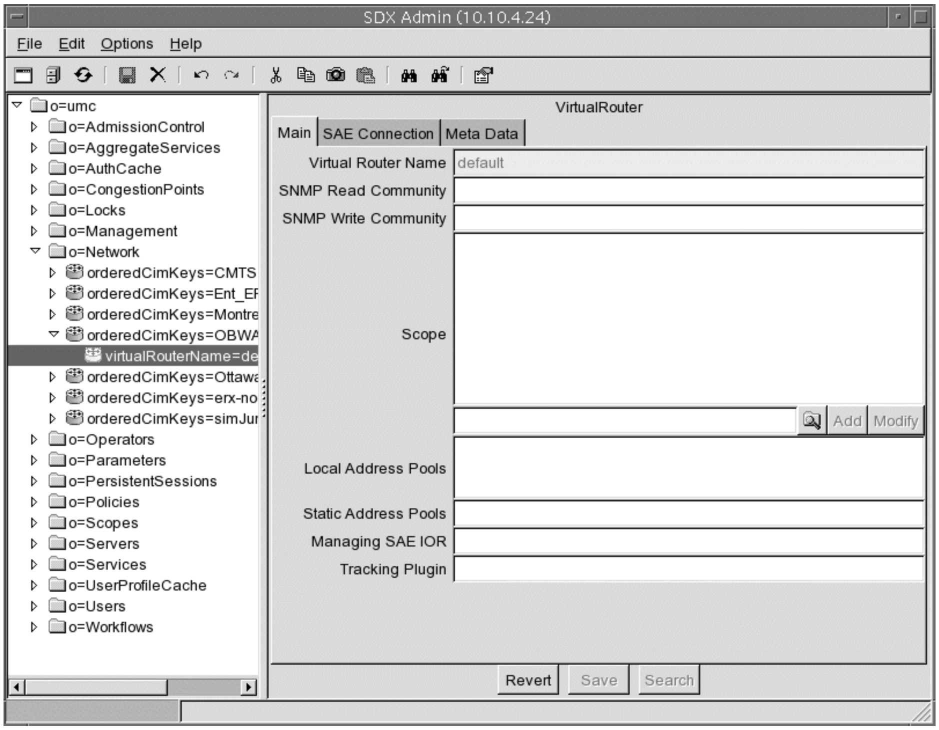
   The New VirtualRouter dialog box appears.

2. Enter the name of the VR, and click **OK**.

   - For JUNOSe routers, the name of the VR, which is case sensitive, must exactly match the name of the VR configured on the router.

   - For JUNOS routing platforms and CMTS devices, use the name default.

   The new VR appears in the navigation pane, and the Main tab of the VirtualRouter pane appears.



Virtual Router Name: default

3. Enter or accept the default values for the virtual router fields.

   See *Virtual Router Fields* on page 120.

4. Select the **SAE Connection** tab in the VirtualRouter pane, and add SAEs that are connected to the router. See *Specifying the SAEs That Can Manage the Router* on page 123.

NOTE: This step is required for the SAE to work with the router.

---

5. Click **Save**.

Virtual Router Fields

In SDX Admin, you can modify the following fields in the content pane for a virtual router (*virtualRouterName = <virtualRouterName> , orderedCimKeys = <EdgeDeviceName> , o = network, o = umc)*.

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.
- Value—Text string
- Example—POP-Westford

Local Address Pools

- List of IP address pools that a JUNOSe VR currently manages and stores.
- Value—You can specify an unlimited number of ranges of local IP address pools for JUNOSe VRs. You can specify either the first and last addresses in a range or the first IP address and a factor that indicates the start of the range. You can also specify IP addresses to exclude. Use spaces in the syntax only to separate the first and last explicit IP addresses in a range.

  The IP pool syntax has the 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>—IP address mask
  - <digitNumber>—Integer specifying the number of significant digits of the first IP address in the range
  - <ipAddressExclude>—List of IP addresses to be excluded from the range
  - |—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
- Guidelines—Configure this field on JUNOSe VRs only. If you do not configure the **PoolPublisher** router initialization scripts for a JUNOSe router, configure this field for the JUNOSe VR.
- Default—No value

- Example—This example shows four ranges for the IP address pool.

  ([10.10.10.5 10.10.10.250]
  {10.20.20.0/24}
  {10.21.0.0/255.255.0.0}
  {10.20.30.0/24,10.20.30.1})

  - The first range (a simple range) specifies all the IP addresses between the two IP addresses 10.10.10.5 and 10.10.10.250.
  - The second range specifies all the IP addresses in the range 10.20.20.0 to 10.20.20.255.
  - The third range uses a network mask to specify all the IP addresses in the range 10.21.0.0 to 10.21.255.255.
  - The fourth range specifies all the addresses of the network 10.20.30.0 to 10.20.30.255, excluding the address 10.20.30.1.

Static Address Pools

- List of IP address pools that a JUNOSe VR manages but does not store. You can configure these address pools only in the SRC software.
- Value—See the field Local Address Pools.
- Guidelines—Configure this field on JUNOSe and CMTS VRs only.
- Default—No value
- Example—([10.10.10.5 10.10.10.250] {10.20.20.0/24})

Managing SAE IOR

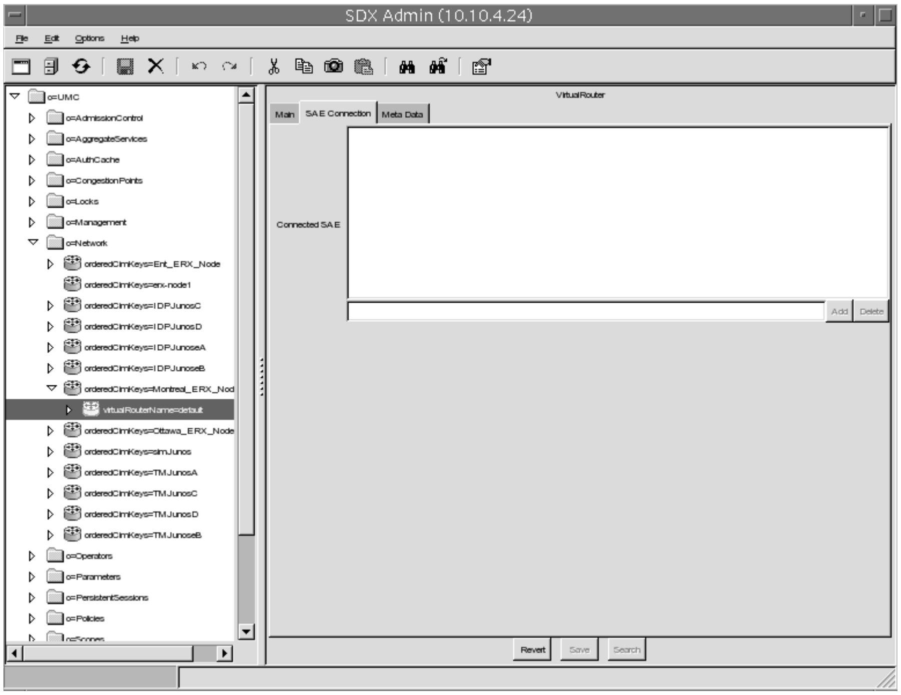
- Common Object Request Broker Architecture (CORBA) reference for the SAE managing this VR.
- Value—One of the following items:
  - The actual CORBA reference for the SAE
  - The absolute path to the interoperable object reference (IOR) file
  - A corbaloc URL in the form corbaloc::<host>:8801/SAE
    - <host> is the name or IP address of the SAE host.
- Default—No value
- Guidelines—The **PoolPublisher** and **IorPublisher** router initialization scripts provide this information when the router connects to the SAE. If you do not select one of these router initialization scripts, enter a value in this field.
- Example—One of the following items:
  - Absolute path—*/opt/UMC/sae/var/run/sae.ior*
  - corbaloc URL—corbaloc::boston:8801/SAE
  - Actual IOR—IOR:000000000000002438444C3A736D67742E6A756E697...

Tracking Plug-in

- Plug-ins that track interfaces that the SAE manages on this VR. The SAE calls these plug-in instances for every interface it manages. 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 network information collector (NIC) SAE plug-in agents that are specific to this VR.
- Default—No value
- Example—nicsae, flexRadius

Specifying the SAEs That Can Manage the Router

You must add the addresses of SAEs that can manage this router. This step is required for the SAE to work with the router. To add the SAEs, select the SAE Connection tab in the VirtualRouter pane.



Adding an SAE

To add an SAE:

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

   To specify the active SAE and the redundant SAE, enter an exclamation point (!) after the hostname or IP address of the connected SAE.

2. Click **Add**.

Modifying an SAE Address

To modify an SAE address:

1. 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**.

Deleting an SAE Address

To delete an SAE address:

1. 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 router or CMTS device.
- Value—IP addresses
- Default—No value

Configuring the SAE to Manage JUNOS Routing Platforms

A JUNOS routing platform interacts with the SAE by using a JUNOS software process called sdxd. When the sdxd process establishes a TCP/IP connection to the SAE, the SAE begins to manage the router. The JUNOS router driver configuration defines parameters related to the interactions between the SAE and the sdxd process. See *Chapter 7, Using JUNOS Routing Platforms in the SRC Network with the SRC CLI*.

Configuring Secure Connections Between the SAE and JUNOS Routing Platforms

You can use TLS to protect communication between the SAE and JUNOS routing platforms.

To complete the handshaking protocol for the TLS connection, the client (JUNOS routing platform) and the server (SAE) must exchange and verify certificates. You need to create a client certificate and a server certificate. Both certificates must be signed by a certificate authority (CA). JUNOS software supports VeriSign, Inc. (http://www.verisign.com). You must then install both certificates on the SAE and on the JUNOS routing platform.

To set up the SAE and the JUNOS routing platform to use TLS, perform the following tasks:

1. Creating a Server Certificate for the SAE on page 125
2. Installing the Server Certificate on the SAE on page 126
3. Installing the Server Certificate on the Router on page 126
4. Creating a Client Certificate for the Router on page 127
5. Installing the Client Certificate on the Router on page 127
6. Installing the Client Certificate on the SAE on page 127
7. Configuring the SAE to Use TLS on page 128
8. Configuring the Keystore for TLS Certificates and Keys on page 128

Creating a Server Certificate for the SAE

The SRC software provides a sample security certificate that you must replace with a real one. You can obtain a signed certificate from a CA. The SAE stores certificates in a keystore, which is a database of keys and certificates from trusted entities.

To remove the sample certificate and create a site certificate:

1. Access the SAE installation directory.

   ```
   cd /opt/UMC/sae
   ```

2. Remove the sample certificate.

   ```
   rm -f lib/jetty/saeKeystore
   ```

3. Generate a self-signed certificate using the **keytool** command; for example:

   ```
   /opt/UMC/jre/bin/keytool -genkey -keyalg RSA -keystore
   keystore/keystore.jks -keypass router -storepass router -alias sae -dname
   <DN> -validity 365
   ```

   The values specified for the **-keystore**, **-keypass**, **-storepass**, and **-alias** arguments must match the following values that you configure for the keystore on the SAE:

   - The value of the **-keystore** argument must match the value of the Keystore Location field.
   - The value of the **-keypass** and **-storepass** arguments must both match the value of the Keystore Password field.

   See *Configuring the Keystore for TLS Certificates and Keys* on page 128.

Replace <DN> with the distinguished name that identifies your HTTPS server. For example, if XYM Corp in Canada has an HTTPS server with a hostname of ssp1.domain.org, then the DN might be:

```
"cn=ssp1.domain.org, o=XYM Corp, c=CA"
```

Be sure to include the quotation marks. Do not use the "#" character in DNs.

For complete documentation of the Java **keytool**, see:

http://java.sun.com/j2se/1.4.1/docs/tooldocs/solaris/keytool.html

4. Create a certificate signing request (CSR).

   ```
   /opt/UMC/jre/bin/keytool -certreq -alias sae -file server.csr -keypass router
   -keystore keystore/keystore.jks -storepass router
   ```

   The command creates a CSR and places it in the *server.csr* file.

5. Send the CSR from the file */opt/UMC/sae/server.csr* for signing to VeriSign, Inc. (http://www.verisign.com).

   VeriSign authenticates you and returns a certificate, signed by them, that authenticates your public key.

Installing the Server Certificate on the SAE

To install the server certificate on the SAE, import the server certificate into the SAE keystore using the **keytool** command:

```
/opt/UMC/jre/bin/keytool -import -alias sae -file server.crt -keypass router
-noprompt -trustcacerts -keystore keystore/keystore.jks -storepass router
```

Installing the Server Certificate on the Router

The TLS client (JUNOS routing platform) needs a copy of the certificate that was used to sign the SAE certificate so that it can verify the SAE certificate. To install the SAE certificate on the JUNOS routing platform:

1. Include the following statements at the [edit security certificates certificate-authority] hierarchy level.

   ```
   [edit security certificates certificate-authority]
   security{
       certificates{
           certificate-authority SAECert{
             File /var/db/certs/cert.pem
           }
       }
   }
   ```

2. Include the following statements at the [system services service-deployment] hierarchy level.

```
system{
    services{
      service-Deployment{
          servers {
            server-address port port-number{
              Security-options {
                tls;
              }
            }
          }
      }
    }
}
```

Creating a Client Certificate for the Router

For information about how to obtain a certificate for the router from a certificate authority, see *Obtaining a Certificate from a Certificate Authority* in the *JUNOS System Basics Configuration Guide.*

Installing the Client Certificate on the Router

To install the client (router) certificate on the JUNOS routing platform:

1. Include the following statements at the [edit security certificates certificate-authority] hierarchy level.

```
[edit security certificates certificate-authority]
security{
    certificates{
          local clientCERT { .... } ;
    }
}
```

2. Include the following statements at the [system services service-deployment] hierarchy level.

```
system{
    services{
          service-Deployment{
              local-certificate clientCert;
          }
    }
}
```

Installing the Client Certificate on the SAE

To install the client certificate on the SAE, you must import the client (router) certificate to the SAE keystore using the **keytool** command. For example:

```
/opt/UMC/jre/bin/keytool -import -alias router -file client.crt -keypass router
-noprompt -trustcacerts -keystore keystore/keystore.jks -storepass router
```

Configuring the SAE to Use TLS

To configure the SAE to accept TLS connections, enter a port number in the TLS BEEP Server Port field in the JUNOS router driver configuration.

See *Configuring the SAE to Manage JUNOS Routing Platforms* on page 124.

Configuring the Keystore for TLS Certificates and Keys

A keystore is a database of keys and certificates from trusted entities. See *Chapter 7, Using JUNOS Routing Platforms in the SRC Network with the SRC CLI*.

Checking Changes to the JUNOS Configuration

The SAE can check the configuration of a JUNOS routing platform under its control to detect whether the configuration has changed by a means other than through the SAE. If the SAE finds a disparity between the router and the SAE configurations, it can take several actions. The SAE checks the configuration installed on the router against the state of the SAE session layer (subscriber, service, and interface sessions). While the check is occurring, the SAE does not handle jobs from the router, and all provisioning activity is blocked, including event notifications.

The SAE can take the following actions if it finds a disparity between the router and SAE configurations:

- Remove the disparate sessions from the router. When the SAE removes a session, it generates Stop events for the session and removes the session from the session store and the SAE.
- Re-create the sessions that have been removed. Subscribers whose sessions have been removed need to log back in before they can activate services. During session re-creation, the SAE responds to event notifications and provisioning operations.

  If the state of the router configuration is lost because of a failover or a restart, it is not possible to re-create the sessions.
- Report disparities to the operator without making any changes to the router configuration.

  The disparities are reported through the SAE router driver event trap called routerConfOutOfSynch and through the info log.

Note that it is not possible to check the consistency of individual provisioning objects. Therefore, modifications to a provisioning object while the SAE is disconnected from the router cannot be detected.

Setting Up Periodic Configuration Checking

To configure the SAE to periodically check the configuration of the JUNOS routing platform, See *Chapter 7, Using JUNOS Routing Platforms in the SRC Network with the SRC CLI*.

Using SNMP to Retrieve Information from JUNOS Routing Platforms

You can use SNMP to retrieve information from the router. For example, if you create a router initialization script that uses SNMP, you need to specify the SNMP communities that are on the router.

We recommend that you specify SNMP communities for each virtual router. (See *Adding Virtual Routers Individually* on page 119.) You can also configure global default SNMP communities.

Configuring Global SNMP Communities in the SRC Software

You can configure global default SNMP communities that are used if a VR does not exist on the router or the community strings have not been configured for the VR. See *Chapter 7, Using JUNOS Routing Platforms in the SRC Network with the SRC CLI*.

Developing Router Initialization Scripts

When the SAE establishes a connection with a router, it can run a router initialization script to customize the setup of the connection. Router initialization scripts are run when the connection between a router and the SAE is established and again when the connection is dropped.

We provide the IorPublisher script in the */opt/UMC/sae/lib* folder. The IorPublisher script publishes the IOR of the SAE in the directory so that a NIC can associate a router with an SAE.

Interface Object Fields

Router initialization scripts interact with the SAE through an interface object called Ssp. The SAE exports a number of fields through the interface object to the script and expects the script to provide the entry point to the SAE.

Table 9 describes the fields that the SAE exports.

Table 9: Exported Fields

Ssp Attribute	Description
Ssp.properties	System properties object (class: java.util.Properties)—The properties should be treated as read-only by the script.
Ssp.errorLog	Error logger—Use the SsperrorLog.printIn (message) to send error messages to the log.
Ssp.infoLog	Info logger—Use the Ssp.infoLog.printIn (message) to send informational messages to the log.
Ssp.debugLog	Debug logger—Use the Ssp.debugLog.printIn (message) to send debug messages to the log.

The router initialization script must set the field Ssp.routerInit to a factory function that instantiates a router initialization object:

- <VRName>—Name of the virtual router in which the COPS client has been configured, format: virtualRouterName@RouterName
- <virtualIp>—Virtual IP address of the SAE (string, dotted decimal; for example: 192.168.254.1)
- <realIp>—Real IP address of the SAE (string, dotted decimal; for example, 192.168.1.20)
- <VRIp>—IP address of the virtual router (string, dotted decimal)
- <transportVR>—Name of the virtual router used for routing the COPS connection, or None, if the COPS client is directly connected

The factory function must implement the following interface:

```
Ssp.routerInit(VRName,
virtualIp,
realIp,
VRIp,
transportVR)
```

The factory function returns an interface object that is used to set up and tear down a connection for a given COPS server. A common case of a factory function is the constructor of a class.

The factory function is called directly after a COPS server connection is established. In case of problems, an exception should be raised that leads to the termination of the COPS connection.

Required Methods

Instances of the interface object must implement the following methods:

- *setup()*—Is called when the COPS server connection is established and is operational. In case of problems, an exception should be raised that leads to the termination of the COPS connection.
- *shutdown()*—Is called when the COPS server connection is terminated to the virtual router. This method should not raise any exceptions in case of problems.

Example: Router Initialization Script

The following script defines a router initialization class named *SillyRouterInit*. The interface class does not implement any useful functionality; it just writes messages to the infoLog when the router connection is created or terminated.

```
class SillyRouterInit:
    def __init__(self, vrName, virtualIp, realIp, vrIp, transportVr):
        """ initialize router initialization object """
        self.vrName = vrName
        Ssp.infoLog.printin("SillyRouterInit created")
    def setup(self):
        """ initialize connection to router """
        Ssp.infoLog.printin("Setup connection to VR %(vrName)s" %
                    vars(self))

    def shutdown(self):
        """ shutdown connection to router """
        Ssp.infoLog.printin("Shutdown connection to VR %(vrName)s" %
                    vars(self))

#
# publish interface object to Ssp core
#
Ssp.routerInit = SillyRouterInit
```

Specifying Router Initialization Scripts on the SAE

To specify router initialization scripts, See *Chapter 7, Using JUNOS Routing Platforms in the SRC Network with the SRC CLI*.

Accessing the Router CLI

You can access the CLIs of Juniper Networks routers from Policy Editor and from SDX Admin through a Telnet or SSH connection. This access allows you to display and change the configuration of the router.

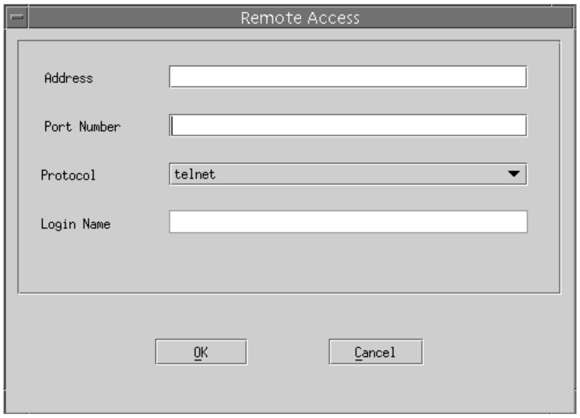
You must have the Telnet or SSH applications installed and available to Policy Editor or SDX Admin. You can open multiple Telnet or SSH sessions.

Using Policy Editor

To access a router from Policy Editor:

1. In the Policy Editor menu, select **Tools > Manage**.

   The Remote Access dialog box appears.



Protocol: telnet

2. Fill in the Remote Access fields, and click **OK**.

   See *Remote Access Fields* on page 133.

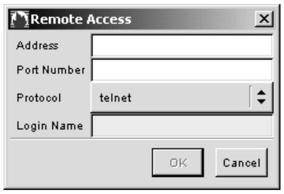
   A Telnet or an SSH window with a CLI prompt appears.

Using SDX Admin

To access a router from SDX Admin:

1. In the navigation pane, expand **o = Network**.
2. Right-click on the router to which you want to connect, and select **Manage**.

   The Remote Access dialog box appears.



Protocol telnet

3. Fill in the Remote Access fields, and click **OK**.

   See *Remote Access Fields* on page 133.

   A Telnet or an SSH window with a CLI prompt appears.

Remote Access Fields

In Policy Editor, you can edit the following fields in the Remote Access dialog box, in the select Tools > Manage menu.

In SDX Admin, you can edit the following fields in the Remote Access dialog box by right-clicking on the router object, and selecting Manage.

Address

- IP address or hostname of the router.
- Value—IP address
- Default—No value
- Example—192.0.2.1

Port Number

- TCP port over which you want to connect to the router.
- Value—TCP port
- Default—No value
- Example—22

Protocol

- Type of connection
- Value—telnet | ssh
- Default—telnet
- Example—ssh

Login Name

- Login name for SSH connections.
- Value—Text string
- Default—No value
- Guideline—You must enter a value for this property.
- Example—admin

Configuring JUNOS Routing Platforms to Interact with the SAE

To configure the JUNOS routing platform to interact with the SAE:

1. Include the following statements at the [edit system services service-deployment] hierarchy level.

```
[edit system services service-deployment]
servers server-address {
port port-number;
}
source-address source-address;
```

2. Use the following guidelines for the variables in these statements.

server-address

- Specifies the IP address of the host on which you install the SAE.
- Value—IP address
- Guidelines—Be sure this setting matches the corresponding value in the SAE configuration.
- Default—None
- Example—192.0.2.2

port-number

- Specifies the port number for the SAE.
- Value—TCP port number
- Guidelines—Be sure this setting matches the corresponding value in the SAE configuration.
- Default—3333
- Example—3333

source-address

- Specifies the IP address of the source that sends traffic to the SAE.
- Value—IP address
- Guidelines—This setting is optional.
- Default—None
- Example—192.0.2.2

Configuring the JUNOS Routing Platform to Apply Changes It Receives from the SAE

To configure the JUNOS routing platform to receive configuration statements from the SAE and apply those statements to the configuration:

1. Create a configuration group called sdx that contains the configuration statements that the SAE sends to the JUNOS routing platform. To do so, include the groups statement at the [edit] level, and specify the name sdx.

   ```
   [edit]
   groups {
       sdx;
   }
   ```

2. Configure the JUNOS routing platforms to apply these statements to the configuration. To do so, include the apply-groups statement at the [edit] level.

   ```
   [edit]
   set apply-groups sdx;
   ```

Disabling Interactions Between the SAE and JUNOS Routing Platforms

To disable the SRC software process, enter the following command.

```
root@ui1#set system processes service-deployment disable
root@ui1#commit
```

When you disable the SRC software process, it is still available on the JUNOS routing platform.

To reenable the SRC software process, enter the following command.

```
root@ui1#delete system processes service-deployment disable
root@ui1#commit
```

The SRC software process attempts to reconnect the JUNOS routing platform to the SAE.

Monitoring Interactions Between the SAE and JUNOS Routing Platforms

Use the following command on JUNOS routing platforms to monitor the connection between the JUNOS routing platform and the SAE.

```
root@ui1> show system services service-deployment
Connected to 172.17.20.151 port 3333 since 2004-02-06 14:50:31 PST
Keepalive settings: Interval 15 seconds
Keepalives sent: 100, Last sent: 6 seconds ago
Notifications sent: 0
Last update from peer: 00:00:06 ago
```

You can also monitor the interactions between the SRC software and JUNOS routing platforms in the log files for the SAE and in the log files generated by the SRC software process on the JUNOS routing platform. For information about configuring logging for the SAE, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 5, Managing SRC Log Files on a Solaris Platform*. For information configuring logging on JUNOS routing platforms, see *JUNOS System Basics Configuration Guide*.

Troubleshooting SRC Problems on JUNOS Routing Platforms

To troubleshoot SRC problems on the JUNOS routing platform, review the log files for the SAE and the log files generated by the SRC software process on the router.

- If the log files indicate that the SRC software process is not responding, see *Troubleshooting Problems with the SRC Software Process* on page 136.
- If the log files indicate a problem with a specific interface, see *Troubleshooting Problems with Interfaces* on page 137.
- If the log files indicate a problem with a specific service or its associated firewall rules, see *Troubleshooting Problems with Services* on page 140.

Troubleshooting Problems with the SRC Software Process

If the log files indicate that the SRC software process is not responding:

1. Look at the status of the process on the JUNOS routing platform.

   ```
   root@ui1>show system services service-deployment
   Connected to 172.17.20.151 port 3333 since 2004-02-06 14:50:31 PST
   Keepalive settings: Interval 15 seconds
   Keepalives sent: 100, Last sent: 6 seconds ago
   Notifications sent: 0
   Last update from peer: 00:00:06 ago
   ```

2. If you see the message “error: the service-deployment subsystem is not running,” reenable the SRC software process (see *Disabling Interactions Between the SAE and JUNOS Routing Platforms* on page 135).
3. If the process is already enabled, review the configurations of the router and the SAE in the directory, and fix any problems.
4. Restart the SRC software process on the router.

   ```
   root@ui1>restart service-deployment
   ```

   The SAE synchronizes with the SRC software process and deletes unnecessary data from the router.

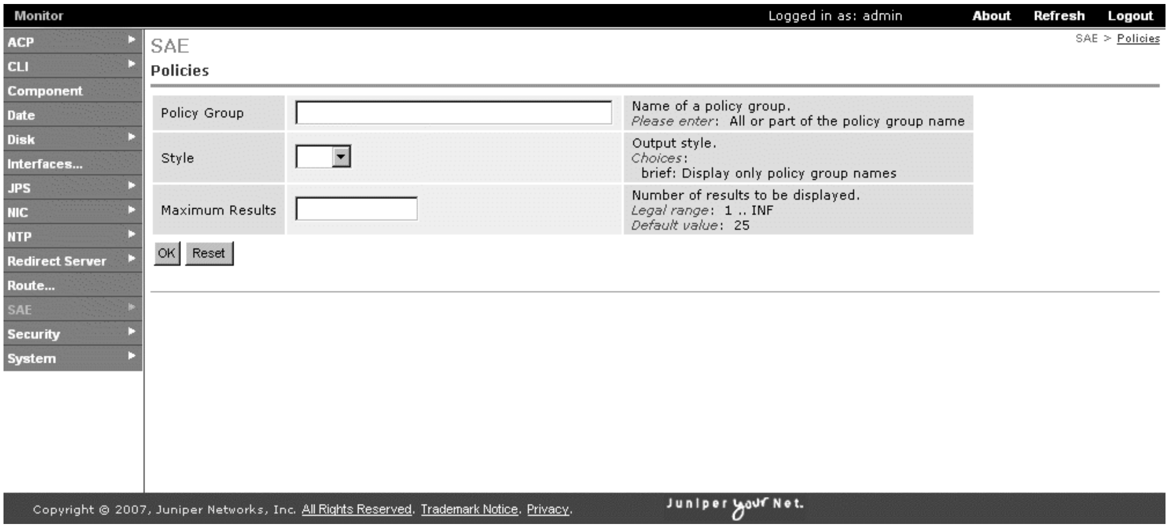
Troubleshooting Problems with Interfaces

If the log files indicate a problem with a specific interface or its associated firewall rules:

1. Review the configuration of the policies associated with the interfaces with the C-Web interface.

   a. Select **SAE** from the side pane, and click **Policies**.

      The Policies pane appears.



Policy Group		Name of a policy group. *Please enter*: All or part of the policy group name
Style		Output style. *Choices*: brief: Display only policy group names
Maximum Results		Number of results to be displayed. *Legal range*: 1 .. INF *Default value*: 25

For more information on these fields, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 16, Monitoring SAE Data with the C-Web Interface*.

   b. Click **OK**.

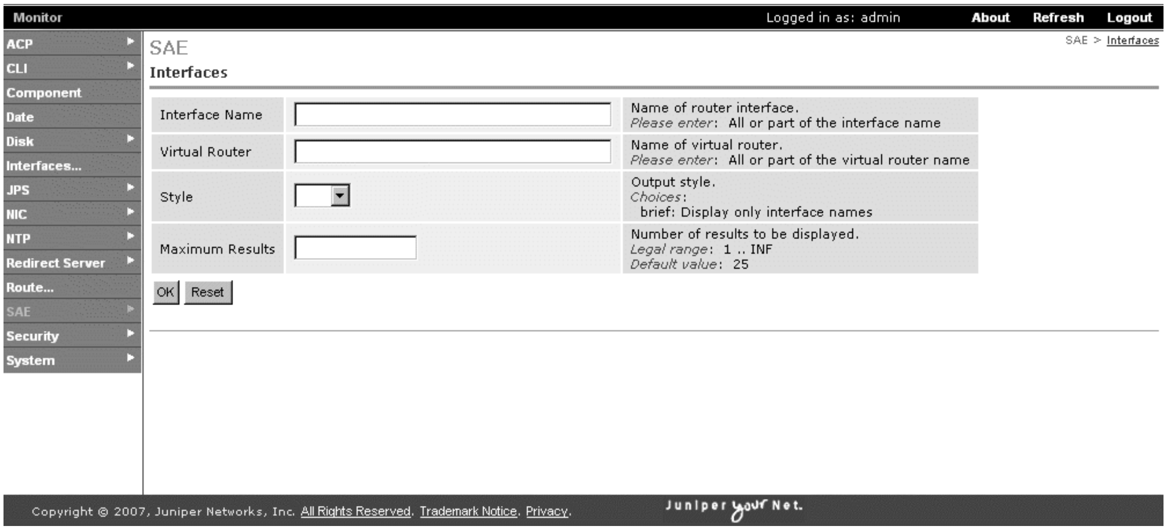
      The Policies pane displays the interfaces available on the router.

   c. If you find any errors, fix the configuration in the directory, and proceed to Step 5. Otherwise, proceed to Step 2.

2. Review the configuration of interfaces on the JUNOS routing platform with the C-Web interface.

   a. Select **SAE** from the side pane, and click **Interfaces**.

      The Interfaces pane appears.



Interface Name	Name of router interface. *Please enter*: All or part of the interface name
Virtual Router	Name of virtual router. *Please enter*: All or part of the virtual router name
Style	Output style. *Choices*: brief: Display only interface names
Maximum Results	Number of results to be displayed. *Legal range*: 1 .. INF *Default value*: 25

   b. Click **OK**.

      The Interfaces pane displays the interfaces available on the router.

   c. If you find any errors, fix the configuration in the directory, and proceed to Step 5. Otherwise, proceed to Step 2.

3. Display the corresponding interfaces on the JUNOS routing platform.

```
root@olive1# show groups sdx interfaces
<fe-0/0/0> {
    unit <0> {
        family inet {
            filter {
                input SDX_PRIVATE_ID0000000000001092282;
                output SDX_PRIVATE_ID0000000000001223352;
            }
        }
    }
}
```

   If you find any errors, fix the configuration in the directory, and proceed to Step 5. Otherwise, proceed to Step 4.

4. Remove the configuration for this interface from the JUNOS routing platform.

    a. Disable the SRC software process.

    ```
    root@ui1#set system processes service-deployment disable
    root@ui1#commit
    ```

    b. Delete the interfaces from the router.

    ```
    delete groups sdx interfaces <interfaceName> <interfaceIdentifier>
    root@ui1#commit
    ```

    For example, to delete the interface with identifier fe-0/0/0 unit 0, enter:

    ```
    root@ui1#delete groups sdx interfaces <fe-0/0/0> unit <0>
    root@ui1#commit
    ```

    c. Reenable the SRC software process.

    ```
    root@ui1#delete system processes service-deployment disable
    root@ui1#commit
    ```

5. Restart the SRC software process on the router.

    ```
    root@ui1>restart service-deployment
    ```

    The SAE reconfigures the interface that you deleted.

6. Review the log files again.

    If the action you took did not fix the problem, return to the last step you performed, and proceed with this troubleshooting procedure. If you have performed all the tasks in the troubleshooting procedure and the problem persists, delete all SRC data on the JUNOS routing platform (see *Deleting All SRC Data on JUNOS Routing Platforms* on page 143*)*.

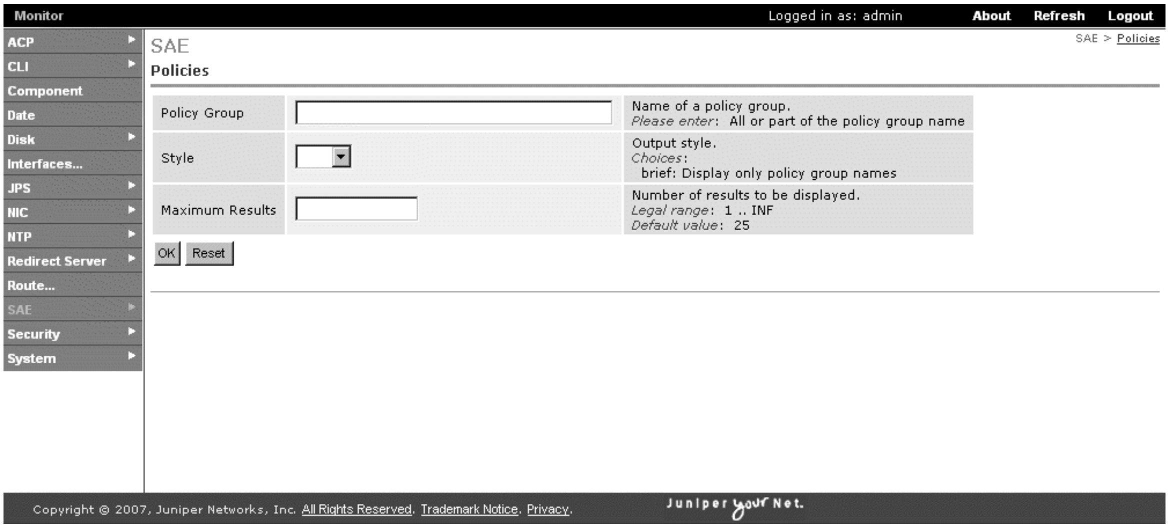
Troubleshooting Problems with Services

If the log files indicate a problem with a specific service or its associated firewall rules:

1. Review the configuration of the policies associated with the interfaces with C-Web.

   a. Select **SAE** from the side pane, and click **Policies**.

      The Policies pane appears.



SAE

Policies

Policy Group		Name of a policy group. *Please enter*: All or part of the policy group name
Style		Output style. *Choices*: brief: Display only policy group names
Maximum Results		Number of results to be displayed. *Legal range*: 1 .. INF *Default value*: 25

   For more information on these fields, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 16, Monitoring SAE Data with the C-Web Interface*.

   b. Click **OK**.

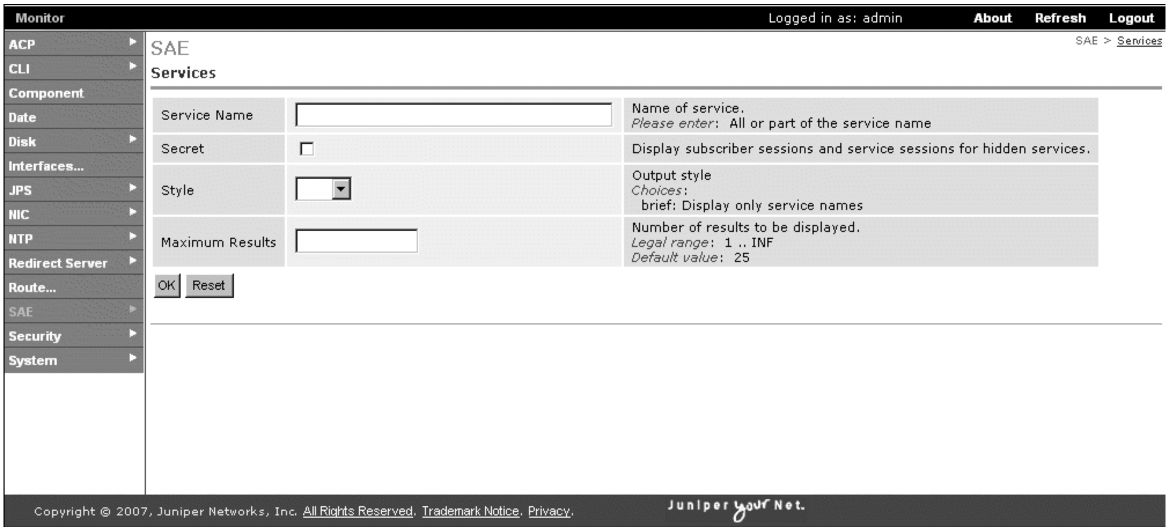
      The Policies pane displays the interfaces available on the router.

   c. If you find any errors, fix the configuration in the directory, and proceed to Step 5. Otherwise, proceed to Step 2.

2. Review the configuration of the service on the JUNOS routing platform with C-Web.

   a. Select **SAE** from the side pane, and click **Services**.

      The Services pane appears.



SAE

Services

Service Name		Name of service. *Please enter*: All or part of the service name
Secret	☐	Display subscriber sessions and service sessions for hidden services.
Style		Output style *Choices*: brief: Display only service names
Maximum Results		Number of results to be displayed. *Legal range*: 1 .. INF *Default value*: 25

For more information on these fields, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 16, Monitoring SAE Data with the C-Web Interface*.

b. Click **OK**.

The Services pane displays the interfaces available on the router.

c. Locate an active service session for this service, and observe the ProvisioningSet field of that session.

d. Locate an identifier that is associated with the service that is causing the problem.

For example, in the above display, the identifier `SDX_PRIVATE_ID0000000000002075317` is associated with a Network Address Translation (NAT) rule.

3. Review the corresponding configuration on the JUNOS routing platform.

   ```
   root@olive1# show groups sdx services nat rule SDX_PRIVATE_ID0000000
   000002075317
       match-direction input;
       term SDX_PRIVATE_TERM {
           from {
               source-address {
                   0.0.0.0/0;
               }
               destination-address {
                   0.0.0.0/0;
               }
           }
           then {
               translated {
                   source-pool SDX_PRIVATE_ID0000000000002009780;
                   translation-type source dynamic;
               }
           }
       }
   ```

   If you find any errors, fix the configuration in the directory and proceed to Step 5. Otherwise, proceed to Step 4.

4. Remove the configuration for this service from the JUNOS routing platform.

   a. Disable the SRC software process.

      ```
      root@ui1#set system processes service-deployment disable
      root@ui1#commit
      ```

   b. Delete the service on the JUNOS routing platform.

      ```
      delete groups sdx services <serviceName> <filterID>
      root@ui1#commit
      ```

      For example, to delete a firewall filter of the service called firewall with filterID SDX_PRIVATE_ID0000000000001223352, enter:

      ```
      delete groups sdx services firewall filter
      SDX_PRIVATE_ID0000000000001223352
      root@ui1#commit
      ```

   c. Reenable the SRC software process.

      ```
      root@ui1#delete system processes service-deployment disable
      root@ui1#commit
      ```

5. Restart the SRC software process on the JUNOS routing platform.

   ```
   root@ui1>restart service-deployment
   ```

   The SAE reconfigures the service that you deleted on the JUNOS routing platform.

6. Review the log files again.

   If the action you took did not fix the problem, return to the last step you performed, and proceed with this troubleshooting procedure. If you have performed all the tasks in the troubleshooting procedure and the problem persists, delete all SRC data on the JUNOS routing platform (see *Deleting All SRC Data on JUNOS Routing Platforms* on page 143*)*.

Deleting All SRC Data on JUNOS Routing Platforms

If deleting parts of the SRC data on a JUNOS routing platform fails to solve problems, delete all the SRC data and restart the SRC software process. To do so:

1. Delete all SRC interfaces and services.

   ```
   delete groups sdx
   root@ui1#commit
   ```

2. If you are running SDX software releases 5.0 through 6.1, you should also delete interface sessions. (After release 6.2, session data is no longer stored on the router, it is stored on the SAE host using the session store feature.)

   ```
   delete groups sdx-sessions
   root@ui1#commit
   ```

3. Restart the SRC software process on the router.

   ```
   root@ui1>restart service-deployment
   ```

   The SAE reconfigures all the interfaces and services that you deleted from the router.

Part 3

Locating Subscriber Management Information

Chapter 9

Locating Subscriber Information with the NIC

This chapter describes the network information collector (NIC) that the SRC software uses to locate subscriber information for an application and discusses strategies for implementing a NIC configuration. The chapter includes information about the NIC sample data provided with the SRC software; reviewing this data will help you plan a NIC configuration for your network. Topics include:

- Locating Subscriber Management Information on page 147
- Mapping Subscribers to a Managing SAE on page 149
- High Availability for NIC on page 150
- Planning a NIC Implementation on page 153
- NIC Agents Used in the NIC Configuration Scenarios on page 156
- Router Initialization Scripts with NIC Configuration Scenarios on page 158

Locating Subscriber Management Information

For services to be activated for a subscriber session, applications such as the SRC Volume-Tracking Application (SRC-VTA), Dynamic Service Activator, Enterprise Manager Portal, or a residential portal need to locate the SAE that manages the subscriber. An application such as the Threat Mitigation Application Portal needs to locate the SAE that manages interfaces through which traffic destined for a specified IP address enters the network.

The NIC is the component that locates which SAE manages a subscriber or an interface. The NIC uses information that identifies the subscriber or the interface to identify the managing SAE. A NIC is similar to a Domain Name System (DNS) in that a NIC processes resolution requests. Rather than translating hostnames to IP addresses and vice versa, the NIC resolves an identifier for a subscriber or an interface to a reference for the managing SAE.

The components that participate in this resolution are a NIC host and a NIC proxy, also called a NIC locator for particular applications. A NIC host processes resolution requests. A NIC proxy requests data resolution for an application. A NIC proxy is so-named because it requests information on behalf of an application. A NIC proxy and a NIC host communicate with each other through Common Object Request Broker Architecture (CORBA); NIC manages the CORBA interactions for you.

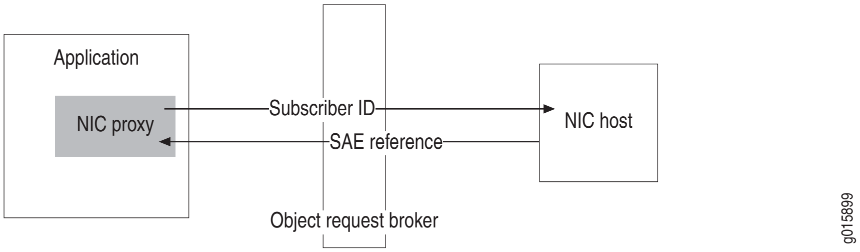
NIC can operate in a client/server mode or in a local host mode. In the client/server mode, a NIC host and NIC proxies can reside on different systems. In local host mode, a NIC host and NIC proxies reside in the same process on a machine.

NIC Client/Server Mode

In client/server mode, a NIC host is the server. A NIC proxy, which comprises libraries within an application that interacts with a NIC host, is the client.

Figure 7 shows a NIC proxy running within an application and a NIC host running on a different machine. Both communicate through CORBA, with the NIC proxy providing an identifier for a subscriber and the NIC host returning a reference to the SAE that manages the subscriber.

Figure 7: Communication Between a NIC Proxy and a NIC Host in Client/Server Mode



NIC Local Host Mode

In local host mode, a Java application can include the libraries for a NIC host as well as NIC proxies. With this configuration, the NIC host and the NIC proxies communicate with each other within the same application. Because both components run within the same application, the application and the NIC host start and stop at the same time.

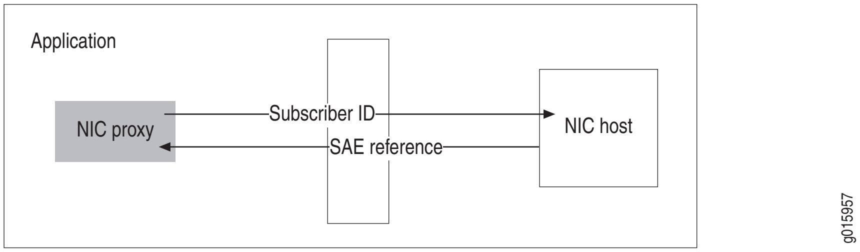
If an application uses a local NIC host, all NIC proxies for the application typically communicate with the local NIC host, but some of the NIC proxies can be configured to communicate with a NIC host that runs on another system.

When you use NIC in local host mode:

- You cannot use C-Web to monitor or troubleshoot the local NIC host
- The NIC host runs all the resolvers and agents for the host on the local machine.
- Other NIC hosts cannot communicate with agents and resolvers that run in a local NIC host.

Figure 8 shows a NIC proxy and a NIC host running within an application.

Figure 8: Communication Between a NIC Host and a NIC Proxy in Local Host Mode



Mapping Subscribers to a Managing SAE

A NIC collects information about the state of the network and can provide mapping from a specified type of network data, known as a *key*, to another type of network data, known as a *value*. Applications can use a NIC proxy to submit a key to a NIC host. The NIC host obtains a corresponding value from other components within NIC and returns it through the NIC proxy to the application. A typical use of a NIC is for a residential portal application to submit a subscriber's IP address and for the NIC to return the interoperable object reference (IOR) of the SAE managing that subscriber.

NIC Proxies and NIC Locators

Typically, an application supports one NIC proxy for each type of data request. A NIC proxy caches resolution results for a period of time so that it can resolve future requests without consulting the NIC host, thereby decreasing traffic between the NIC proxy and the NIC host. Applications that use NIC proxies communicate with the proxy to delete any invalid cache entries. Caching lets you optimize resolution performance for your network configuration and system resources.

You configure a NIC proxy when you configure that application. SRC applications such as the SRC-VTA and Dynamic Service Activator contain NIC proxies. If you are writing an external application that will interact with a NIC, you must include NIC proxies in the application.

A NIC locator provides the same functionality as a NIC proxy; however, it runs as part of the NIC host. A NIC locator uses the NIC access interface module, a simple CORBA interface, to enable non-Java applications to interact with NIC. A NIC locator does not cache information.

For information about the NIC access interface module, see the API documentation in the SRC software distribution in the folder *SDK/doc/idl/nic* or on the Juniper Networks Web site at

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

For more information about NIC proxies and NIC locators, see *Chapter 16, Developing Applications That Use NIC.*

NIC Hosts

NIC hosts collect and store SRC information, and respond to requests from NIC proxies. The components in a NIC host that manage this process are:

- NIC agents—Collect data from SRC components, publish data, and make data available to NIC resolvers
- NIC resolvers—Process resolution requests

NIC Agents

NIC agents collect information about the state of the network from many data sources on the network. Table 10 describes the types of agents supplied with NIC.

Table 10: Types of NIC Agents

Type of Agent	Type of Information the Agent Makes Available
Consolidator agent	Summary information received from other agents.
Directory agent	Specified directory entries and changes to directory entries.
Properties agent	Information from a specified list of property file. Typically, you do not configure properties agents.
SAE plug-in agent	Subscriber information and interface information for SAE-managed subscribers and interfaces.
XML agent	Information from a specified XML document. Typically, you do not configure XML agents.

NIC Resolvers

NIC resolvers manage information to resolve requests by:

- Receiving and storing information about the state of the network from components within NIC and other NIC resolvers
- Requesting information from NIC agents and other NIC resolvers
- Receiving requests from the NIC proxies or other NIC resolvers
- Processing requests and sending responses to the requesters

High Availability for NIC

You can configure high availability for NIC when you use client/server mode with the NIC host and the NIC proxies running on different machines. NIC supports several mechanisms to maintain high availability. We recommend that you use NIC replication to keep a NIC configuration highly available. NIC replication uses groups of NIC hosts that share the same configuration for NIC resolutions to respond to resolution requests.

When you use NIC in local host mode, you do not need to configure redundancy for a NIC host, because the NIC host runs within the application.

High Availability in Existing NIC Configurations

If you have a previous NIC configuration, you may be using:

- NIC host redundancy, in which a set of NIC hosts provide redundancy

  The SRC CLI does not support NIC host redundancy.

- Redundancy for SAE plug-in agents, in which a set of SAE plug-in agents provide redundancy

  If you have an SAE plug-in agent that uses agent redundancy, enable state synchronization for the agent and use NIC replication. In SRC Release 1.0.0, configuration for SAE plug-in agent redundancy is discontinued.

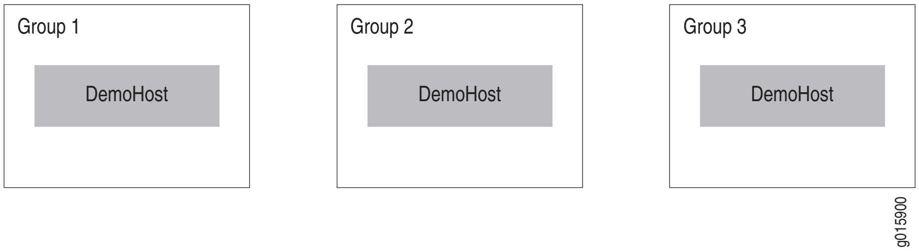
NIC Replication

NIC replication uses the concept of a group to identify a NIC host that has a particular configuration. A group contains one or more NIC hosts; each NIC host in a group is unique; for example, each NIC host could reside on a different system. A NIC proxy contacts specified groups that contain hosts with the same configuration to locate a managing SAE.

For example, a group might include the host DemoHost, but not two instances of DemoHost. Typically, each NIC host in a group is located in the same point of presence (POP). However, a machine can support only one NIC host. The SRC software stores groups in the directory in *ou=dynamicConfiguration, ou=Configuration, o=Management, o=umc*.

For example, Figure 9 shows three NIC groups with each group containing a NIC host that has the same configuration.

Figure 9: NIC Groups



Groups let you:

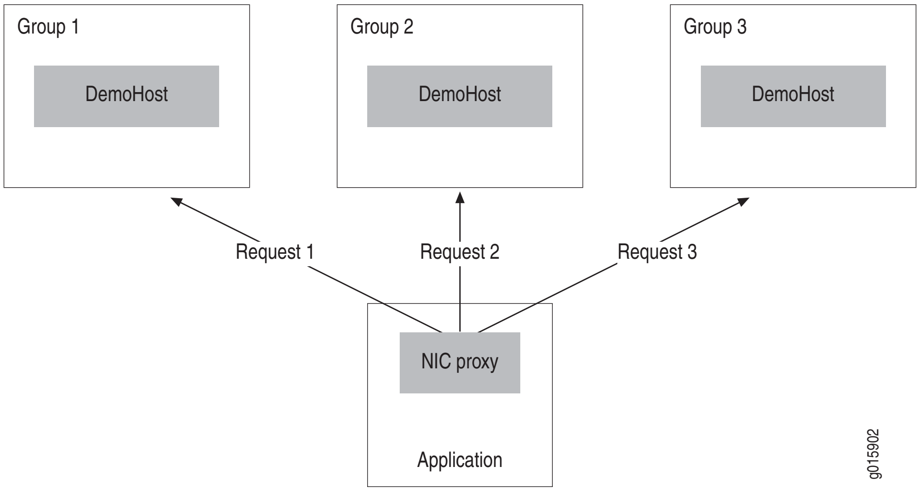
- Distribute network and processing load between two or more groups
- Provide failover protection if one group becomes unavailable

With NIC replication, a NIC proxy can contact multiple NIC hosts that are assigned to different groups. When a NIC proxy is configured to contact more than one group, the NIC configuration on a NIC host in each group should be equivalent—the NIC hosts should use the same configuration scenarios.

A NIC proxy selects a group by using the method specified in the configuration for the proxy; for example, the NIC proxy can randomly choose a group from a list. The NIC proxy then sends resolution requests to the corresponding host in that group. If a NIC proxy submits high numbers of resolution requests to the NIC host, you can configure the NIC proxy to randomly pick a NIC host or to pick a NIC host in a cyclic order to decrease the probability that one NIC host manages all the resolution requests.

Figure 10 shows resolution requests sent by means of a round-robin selection.

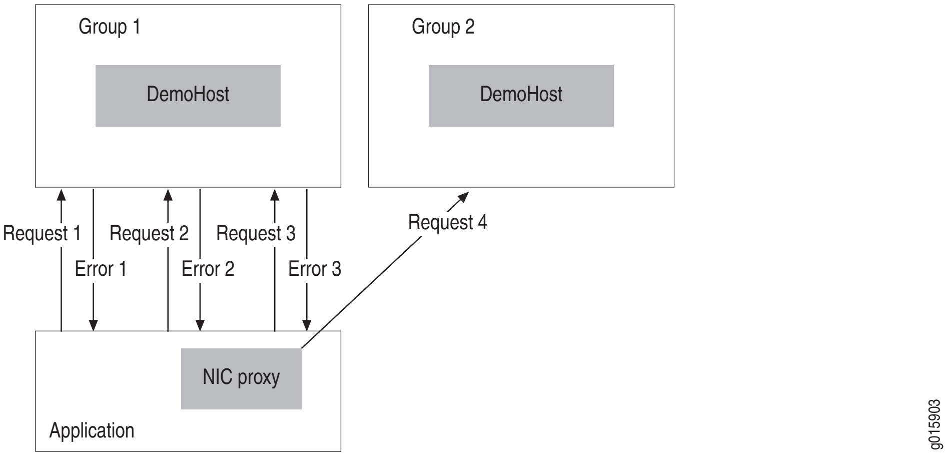
Figure 10: NIC Group Selection by Round-Robin



If the NIC host fails to respond to a specified number of resolution requests, the NIC proxy stops sending resolution requests to the unavailable NIC host and sends the resolution requests to another NIC host. The NIC proxy continues to poll the unavailable NIC host to determine its availability. When the NIC host becomes available, the NIC proxy can again send resolution requests to that host.

Figure 11 shows a NIC proxy that sends a resolution request to Group 1, receives an error message, then sends two more resolution requests before sending a request to Group 2 rather than Group 1. When Group 1 is available again, the NIC proxy will send the request to Group 1.

Figure 11: NIC Resolution Request



You configure NIC replication for hosts, then configure NIC proxies to use replication.

Although you can distribute agents and resolvers among different hosts, as shown in the configuration for the NIC hosts OnePopBO and OnePopH1 in the sample data, we recommend that you use the DemoHost configuration, which centralizes the configuration for agents and resolvers.

Planning a NIC Implementation

The SRC software provides standard NIC configuration scenarios that you can modify to meet the requirements for your environment. Which scenarios you choose depends on the applications you use.

If the resolution scenarios do not provide the type of resolution needed, we recommend that you consult Juniper Professional Services.

If you want to customize configuration of the scenarios provided for a NIC running on a Solaris platform, see *Chapter 18, Customizing a NIC Configuration*.

To plan your NIC implementation:

1. Review the NIC configuration scenarios, and select the scenario that best fits the requirements for your application. In most cases, one of the basic configuration scenarios provides the type of resolution needed.

   See *NIC Configuration Scenarios* on page 154.

2. Determine the number of NIC proxies that you will need to access NIC hosts, and estimate the amount of traffic between the NIC proxies and the NIC hosts. If you expect heavy traffic between NIC proxies and NIC hosts, configure a number of NIC hosts to share the traffic load and processing.

3. Determine which NIC hosts to assign to a group to provide NIC replication; choose names for these groups.

4. If you have not done so already, determine which systems are to run NIC hosts.

NIC Configuration Scenarios

Table 11 lists the NIC configuration scenarios provided in the SRC software.

Table 11: NIC Configuration Scenarios

Configuration Scenario	Name of NIC Configuration Scenario to Use	Type of Resolution	Notes
Basic Configuration Scenarios			
For JUNOSe local configuration for PPP and DHCP subscribers. **Sample use:** DSL providers for residential customers.	OnePop	Subscriber IP address to the SAE IOR	Simplest configuration. IP pools configured locally on each virtual router (VR) with IP addresses from a static pool of IP addresses configured on the virtual router.
For subscribers who have an accounting ID. Can be used for multiple subscribers who use the same accounting ID, in which case NIC returns all SAE IORs for mapped subscribers. **Sample use:** Support for the volume-tracking application.	OnePopAcctId	Accounting ID of a subscriber to the SAE IOR and the IP address of a subscriber to accounting ID	A subscriber's accounting ID can be specified at subscriber login from the SAE subscriber classification script. As a result, the accounting ID encapsulates other attributes of the subscriber session processed by the subscriber classification script. The OnePopAcctId configuration scenario can resolve the encapsulated attributes. For example, customers can assign a subscriber username (login id without domain name) to an accounting ID with the following subscriber classification. [<-retailerDn->?accountingUserId=<-userName->?sub?(uniqueID=<-userName->)]
For subscribers who have assigned IP addresses (assigned external to the SAE). **Sample use:** In a PacketCable Multimedia Specification (PCMM) environment when the SAE acts as both a policy server and application manager.	OnePopDynamicIp	Subscriber IP address to the SAE IOR	

Table 11: NIC Configuration Scenarios (continued)

Configuration Scenario	Name of NIC Configuration Scenario to Use	Type of Resolution	Notes
For resolution of a subscriber login name to an SAE IOR, and of a subscriber IP address to a subscriber login name. **Sample use:** Support for tracking subscriber bandwidth usage or for using a billing model. You can use the SRC-VTA with this scenario.	OnePopLogin	Subscriber login name to the SAE IOR and subscriber IP address to login name	Uses two resolvers. Use a separate NIC proxy for each resolution.
For subscribers who connect through a cable modem termination system (CMTS) device. **Sample use:** In a PCMM environment in which the policy server is separate from the application server. This scenario can be used when the configuration includes Juniper Policy Server or another policy server, and the SAE is an application manager.	OnePopPcmm	Subscriber IP address to the SAE IOR	
For use with applications that use the SAE programming interfaces and that identify subscribers by the primary username. **Sample uses:** ■ Aggregate services ■ Dynamic service activator application	OnePopPrimaryUser	Primary username of a subscriber to the SAE IOR	Similar to *OnePopLogin.xml.*
For a router configuration in which VRs share IP pools. **Sample use:** ■ Services for enterprise subscribers. ■ Support for two different proxies: ■ Subscriber DN to the SAE IOR ■ Subscriber IP address to the SAE IOR	OnePopDnSharedIp	Subscriber distinguished name (DN) or subscriber IP address to the SAE IOR	Includes resolution available in *OnPopSharedIp.xml* and adds resolution from a subscriber DN.

Table 11: NIC Configuration Scenarios (continued)

Configuration Scenario	Name of NIC Configuration Scenario to Use	Type of Resolution	Notes
For a router configuration in which pools can be shared among routers. Pools can be assigned by RADIUS or by a DHCP server. **Sample use:** Support for DHCP and PPP connections for residential subscribers.	OnePopSharedIp	Subscriber IP address to the SAE IOR	
For scenarios in which subscribers have an assigned IP address and these IP addresses can be associated with interfaces on JUNOS routing platforms. **Sample use:** ■ Threat Mitigation Application Portal	OnePopStaticRouteIp	Assigned subscriber IP address to the SAE IOR	Static route information for routers resides in an XML document in the directory under the router object.
For enterprise customers.	OnePopAllRealms	Subscriber IP address or subscriber DN to the SAE IOR	The scenario combines the OnePop and OnePopSharedIp scenarios and adds resolution from a subscriber DN.
Advanced Configuration Scenario			
For two POPs that share a back office. **Sample use:** Support for a deployment that has a back office that connects to NIC hosts at other sites.	MultiPop	Subscriber IP address to the SAE IOR	You can deploy this scenario in an environment that has a number of POPs; for example, a configuration in which there are two POPS with NIC proxy communication to a back office, which in turn communicates with the POP hosts. The POP hosts each support parallel hosts and agents and manage resolutions in the same way. You can add POPs by copying the configuration for one POP and modifying the configuration to suit your environment.

NIC Agents Used in the NIC Configuration Scenarios

When you configure a NIC configuration scenario, you use the basic configuration for each NIC agent in the scenario, but modify properties such as directory properties to make the agent configuration compatible with your SRC configuration. The NIC configuration scenario that you use determines which agents appear in your configuration.

Table 12 lists all agents that are available in the various configuration scenarios.

Table 12: NIC Agents

Agent Name	Type of Agent	Type of Information
AcctIdIp	SAE plug-in	Mappings of accounting IDs of a subscribers to the SAE IOR and subscriber IP addresses to accounting ID(s)
DnVr	SAE plug-in	Mappings of enterprise access DNs to VRs
Enterprise	Directory	List of enterprise names
IpAcctId	SAE plug-in	Mappings of subscriber IP addresses to accounting IDs
IpLoginName	SAE plug-in	Mappings of IP addresses to login names
IpVr	SAE plug-in	Mappings of IP addresses to VRs
LoginNameVr	SAE plug-in	Mappings of login names to VRs
PoolVr	Directory	Mappings of IP pools to VRs
UserNameVr	SAE plug-in	Mappings of subscriber IP addresses to accounting IDs
VrSaeId	Directory	Reads information about virtual routers and the mappings between virtual routers and SAEs

Table 13 shows the types of agents that each configuration scenario uses.

Table 13: Agents in Configuration Scenarios

NIC Configuration Scenario	Directory Agents	SAE Plug-In Agents
OnePop	PoolVr, VrSaeId	
OnePopAcctId	PoolVr, VrSaeId	AcctIdIp, IpAcctId
OnePopDnSharedIp	PoolVr, VrSaeId, Enterprise	DnVr
OnePopDynamicIp	PoolVr, VrSaeId	
OnePopLogin	Pool, VrSaeId	IpLoginName, LoginNameVr
OnePopPcmm	PoolVr, VrSaeId	
OnePopSharedIp	PoolVr, VrSaeId	IpVr
MultiPop	PoolVr, VrSaeId, site-specific versions of PoolVr and VrSaeId	IpVr
OnePopAllRealms	PoolVr, VrSaeId, Enterprise	IpVr
OnePopPrimaryUser	VrSaeId	UserNameVr
OnePopStaticRouteIp	VrSaeId, PoolInterface	

NOTE: If you use a configuration scenario that includes an SAE plug-in agent, make sure that your network has a CORBA naming server that includes the names of the servers that host the SAE plug-in agents. The SRC software distribution includes a CORBA naming server in the omniORB package.

Router Initialization Scripts with NIC Configuration Scenarios

The NIC resolutions map VRs to SAEs. For these resolutions, use a router initialization script that associates each VR with the SAE that manages it. Which router initialization script you use depends on whether the SAE obtains IP pools from JUNOSe VRs:

- **poolPublisher** router initialization script—Use when the SAE obtains local IP pools locally from JUNOSe VRs.
- **iorPublisher** router initialization script—Use when the router is one of the following:
    - JUNOSe routers that do not supply IP addresses from local pools
    - JUNOS routing platforms
    - CMTS devices

  These devices do not supply IP addresses from local pools in your network.

Table 14 lists which type of initialization script should be used with the various NIC configuration scenarios.

Table 14: Type of Router Initialization Script to Use for NIC Configuration Scenarios

poolPublisher	iorPublisher	poolPublisher or iorPublisher
One Pop	OnePopDnSharedIp	OnePopAcctId
	OnePopPcmm	OnePopAllReams
	OnePopPrimaryUser	OnePopDynamicIp
	OnePopSharedIp	OnePopLogin
	OnePopStaticRouteIp	MultiPop

NOTE: If you modify information about IP pools on a VR after the COPS connection is established, the SAE does not automatically register the changes, and you must update the directory.

For more information about router initialization scripts for JUNOSe routers, including how to update the directory, see *Chapter 5, Using JUNOSe Routers in the SRC Network with the SRC CLI*.

For more information about router initialization scripts for JUNOS routing platforms, see *Chapter 7, Using JUNOS Routing Platforms in the SRC Network with the SRC CLI.*

Chapter 10

Configuring NIC with the SRC CLI

This chapter describes how you can use the SRC CLI to configure the network information collector (NIC). You can use the CLI to configure the NIC on a Solaris platform or on a C-series Controller.

Topics in this chapter include:

Configuration Statements for the NIC

The SRC CLI provides the following groups of configuration statements for the NIC:

- *Configuration Statements for NIC Operating Properties* on page 160
- *Configuration Statements for NIC Scenarios* on page 161
- *Configuration Statements for NIC Logging* on page 162

NOTE: We recommend that you change only those statements visible at the basic editing level. Contact Juniper Professional Services or Juniper Customer Support before you change any of the NIC statements and options not visible at the basic editing level.

Configuration Statements for NIC Operating Properties

Use the following configuration statements to configure the NIC operating properties at the [edit] hierarchy level. These statements are visible at the CLI basic editing level.

```
slot number nic {
    base-dn base-dn;
    java-heap-size java-heap-size;
    snmp-client;
    hostname hostname;
    runtime-group runtime-group;
}

slot number nic initial {
    static-dn static-dn;
    dynamic-dn dynamic-dn;
}

slot number nic initial directory-connection {
    url url;
    backup-urls [backup-urls...];
    principal principal;
    credentials credentials;
    protocol (ldaps);
    timeout timeout;
    check-interval check-interval;
    blacklist;
    snmp-agent;
}
```

```
slot number nic initial directory-eventing {
    eventing;
    signature-dn signature-dn;
    polling-interval polling-interval;
    event-base-dn event-base-dn;
    dispatcher-pool-size dispatcher-pool-size;
}
```

For detailed information about each configuration statement, see the *SRC-PE CLI Command Reference.*

Configuration Statements for NIC Scenarios

Use the following configuration statements to configure the NIC at the [edit] hierarchy level. These statements are visible at the CLI basic editing level.

Which agents you configure depends on the NIC configuration scenario that you use.

NOTE: The CLI also provides configuration statements for consolidator agents, properties agents, and XML agents. At this time, none of the NIC configuration scenarios uses these agents. The following list does not include the configuration statements for these agents.

```
shared nic scenario name

shared nic scenario name agents name

shared nic scenario name agents name configuration directory {
    search-base search-base;
    search-filter search-filter;
    search-scope (0 | 1 | 2);
    server-url server-url;
    directory-backup–urls directory-backup-urls;
    principal principal;
    credentials credentials;
}

shared nic scenario name agents agent configuration sae-plug-in {
    event-filter event-filter;
    number-of-events number-of-events;
}
```

For detailed information about each configuration statement, see the *SRC-PE CLI Command Reference.*

Configuration Statements for NIC Logging

Use the following configuration statements to configure logging for the NIC at the [edit] hierarchy level.

```
shared nic scenario name hosts name configuration logger name syslog {
    filter filter;
    host host;
    facility facility;
    format format;
}

shared nic scenario name hosts name configuration logger name file {
    filter filter;
    filename filename;
    rollover-filename rollover-filename;
    maximum-file-size maximum-file-size;
}
```

Before You Configure the NIC

When you use NIC in a client/server configuration, you configure the NIC scenario before you configure the NIC proxies.

Before you configure NIC hosts from the CLI:

- Plan your NIC implementation:
    - Choose the NIC configuration scenario to use.

      The default scenario is OnePop.

      For information about NIC configuration scenarios and NIC agents, see *Chapter 9, Locating Subscriber Information with the NIC*.
- Ensure that the appropriate type of router initialization script is configured for the router or network device.

  See *Chapter 9, Locating Subscriber Information with the NIC.*
- Set the editing level for the CLI to basic. This ensures that only the statements that you need to configure are visible.

  See *SRC-PE CLI User Guide, Chapter 9, Controlling the SRC CLI Environment.*

Configuring the NIC with the SRC CLI

Before you configure the NIC, complete the prerequisite tasks.

See *Before You Configure the NIC* on page 162.

To configure the NIC:

1. Start the NIC component.

   See *Starting the NIC with the SRC CLI* on page 164.

2. Configure NIC operating properties.

   See *Reviewing and Changing Operating Properties for NIC with the SRC CLI* on page 164.

3. Configure NIC replication.

   See *Configuring NIC Replication with the SRC CLI* on page 166.

4. (Optional) If you plan to use a configuration scenario other than OnePop (the default), delete any data for the OnePop scenario and configure the local static DN to specify the configuration scenario.

   See *Changing NIC Configurations with the SRC CLI* on page 179.

5. Configure a NIC scenario.

   See *Configuring a NIC Scenario with the SRC CLI* on page 167.

6. Verify the NIC configuration.

   See *Verifying Configuration for the NIC with the SRC CLI* on page 176.

Related Topics

- *Chapter 9, Locating Subscriber Information with the NIC*
- *Changing NIC Configurations with the SRC CLI* on page 179
- *Testing a NIC Resolution with the SRC CLI* on page 176

Starting the NIC with the SRC CLI

Start the NIC component before you configure it. When you enable NIC for the first time, it creates the default operating properties for the component.

To start NIC:

- From operational mode, enable the NIC.

  ```
  user@host> enable component nic
  Starting NICHOST: may take a few minutes...
  ```

Related Topics

- *Configuring the NIC with the SRC CLI* on page 163

Reviewing and Changing Operating Properties for NIC with the SRC CLI

Before you configure a NIC configuration scenario, review the default operating properties and change values as needed. Operating properties are configured for a slot.

Reviewing the Default NIC Operating Properties

To review the default NIC operating properties:

1. From configuration mode, access the configuration statement that specifies the configuration for the NIC on a slot.

   ```
   [edit]
   user@host# edit slot number nic
   ```

   For example:

   ```
   [edit]
   user@host# edit slot 0 nic
   ```

2. Run the show command.

   ```
   [edit slot 0 nic]
   user@host# show
   base-dn o=umc;
   java-runtime-environment ../jre/bin/java;
   java-heap-size 128m;
   snmp-agent;
   hostname DemoHost;
   initial {
     dynamic-dn "ou=dynamicConfiguration, ou=Configuration,
   o=Management,<base>";
     directory-connection {
       url ldap://127.0.0.1:389/;
       backup-urls ;
       principal cn=nic,ou=Components,o=Operators,<base>;
       credentials ********;
       timeout 10;
   ```

```
    check-interval 60;
  }
  directory-eventing {
    eventing;
    signature-dn <base>;
    polling-interval 15;
    event-base-dn <base>;
    dispatcher-pool-size 1;
  }
  static-dn "l=OnePop,l=NIC, ou=staticConfiguration, ou=Configuration,
o=Manage
ment,<base>";
}
```

Changing NIC Operating Properties

In most cases you can use the default NIC operating properties. Change the default properties if needed for your environment.

To change NIC operating properties:

1. From configuration mode, access the configuration statement that specifies the configuration for the NIC on a slot.

   [edit]
   user@host# **edit slot** number **nic**

   For example:

   [edit]
   user@host# **edit slot 0 nic**

2. (Optional) If you store data in the directory in a location other than the default, *o=umc*, change this value.

   [edit slot 0 nic]
   user@host# **set base-dn** *base-dn*

3. (Optional) If you determine that additional memory is needed, change the maximum memory size available to the (Java Runtime Environment) JRE.

   [edit slot 0 nic]
   user@host# **set java-heap-size** *java-heap-size*

   By default, the JRE can allocate 128 MB. Set to a value lower than the available physical memory to avoid low performance because of disk swapping.

   If you use an SAE plug-in agent, we recommend that you increase the JVM max heap to a value in the range 400–500 MB.

   If you need help to determine the amount of memory needed, contact Juniper Networks Customer Services and Support.

4. (Optional) Enable viewing of SNMP counters through an SNMP browser.

   ```
   [edit slot 0 nic]
   user@host# set snmp-agent
   ```

5. (Optional) Change the name of the NIC host. Use the default name of the NIC host configured for a NIC scenario. In most cases, the NIC host name is DemoHost.

   ```
   [edit slot 0 nic]
   user@host# set hostname hostname
   ```

6. (Optional) Change the initial properties.

   See *SRC-PE Getting Started Guide, Chapter 30, Configuring Local Properties with the SRC CLI*.

Related Topics

- *Configuring the NIC with the SRC CLI* on page 163

Configuring NIC Replication with the SRC CLI

You configure NIC replication to keep the NIC configuration highly available.

Before you configure NIC replication:

- Make sure that you understand how NIC groups are used.

  See *Chapter 9, Locating Subscriber Information with the NIC*.

- Identify which NIC hosts are to provide redundancy for each other.
- Select a name for a group for each of these hosts.

To configure NIC replication:

1. From configuration mode, access the configuration statement that specifies the configuration for the agent.

   ```
   [edit]
   user@host# slot number nic
   ```

   For example:

   ```
   [edit]
   user@host# slot 0 nic
   ```

2. Configure the runtime group for the NIC host.

   ```
   [edit slot 0 nic]
   user@host# runtime-group runtime-group
   ```

   For example:

   ```
   [edit slot 0 nic]
   user@host# runtime-group group1
   ```

Related Topics

- *Configuring the NIC with the SRC CLI* on page 163

Configuring a NIC Scenario with the SRC CLI

The OnePop configuration scenario is the default configuration for NIC. If you want to use another configuration scenario, you first clear data for the configuration scenario and change the static DN that identifies the scenario, see *Changing NIC Configurations with the SRC CLI* on page 179.

When you select a NIC configuration scenario, the software adds the default configuration for most properties. You can modify the NIC properties, including those for agents.

CAUTION: We recommend that you change only those statements visible at the basic editing level. Contact Juniper Professional Services or Juniper Customer Support before you change any of the NIC statements not visible at the basic editing level.

To specify a NIC configuration scenario for NIC to use:

1. Make sure that the NIC component is running.

   ```
   user@host> show component
   Installed Components
   Name      Version                                            Status
   ...
   nic       Release: 7.0 Build: GATEWAY.A.7.0.0.0168           running
   ...
   ```

2. From configuration mode, access the statement that configures a NIC configuration scenario, and specify the name of a scenario.

   ```
   [edit]
   user@host# edit shared nic scenario name
   ```

   For example:

   ```
   [edit]
   user@host# edit shared nic scenario OnePopLogin
   ```

3. View the default configuration for the configuration scenario. For example:

```
[edit shared nic scenario OnePopLogin]
user@host# show

hosts {
  DemoHost {
    configuration {
      hosted-resolvers "/realms/login/A1, /realms/login/B1,
/realms/login/C1, /realms/login/D1, /realms/ip/A1, /realms/ip/B1,
/realms/ip/C1";
      hosted-agents "/agents/LoginNameVr, /agents/VrSaeId,
/agents/IpLoginName,
/agents/PoolVr";
    }
  }
  OnePopBO {
    configuration {
      hosted-resolvers "/realms/login/A1, /realms/login/C1, /realms/ip/A1,
/real
ms/ip/C1";
      hosted-agents /agents/VrSaeId;
    }
  }
  OnePopH1 {
    configuration {
      hosted-resolvers "/realms/login/B1, /realms/login/D1, /realms/ip/B1";
      hosted-agents "/agents/LoginNameVr, /agents/IpLoginName,
/agents/PoolVr";
    }
  }
}
agents {
  VrSaeId {
    configuration {
      directory {
        search-base o=Network,<base>;
        search-filter (objectclass=umcVirtualRouter);
        search-scope 2;
        server-url ldap://127.0.0.1:389/;
        backup-servers-url ;
        principal cn=nic,ou=Components,o=Operators,<base>;
        credentials ********;
      }
    }
  }
  LoginNameVr {
    configuration {
      sae-plug-in {
        event-filter "(&(!(PA_USER_TYPE=INTF))(!(PA_LOGIN_NAME=[None])))";
        number-of-events-sent-in-a-synchronization-call 50;
      }
    }
  }
  IpLoginName {
    configuration {
      sae-plug-in {
number-of-events-sent-in-a-synchronization-call 50;
      }
    }
  }
  PoolVr {
    configuration {
      directory {
        search-base o=Network,<base>;
        search-filter (objectclass=umcVirtualRouter);
        search-scope 2;
        server-url ldap://127.0.0.1:389/;
        backup-servers-url ;
        principal cn=nic,ou=Components,o=Operators,<base>;
        credentials ********;
      }
    }
  }
}
```

4. (Optional) Update logging configuration.

   See *SRC-PE Monitoring and Troubleshooting Guide, Chapter 3, Configuring Logging for SRC Components with the CLI*.

   By default, NIC has the following logging enabled for a NIC host:

   ```
   logger file-1 {
       file {
         filter !ConfigMgr,!DES,/debug-;
         filename var/log/nicdebug.log;
         rollover-filename var/log/nicdebug.alt;
         maximum-file-size 10000000;
       }
     }
     logger file-2 {
       file {
         filter /info-;
         filename var/log/nicinfo.log;
       }
     }
     logger file-3 {
       file {
         filter /error-;
         filename var/log/nicerror.log;
   ```

5. For each agent that the NIC configuration scenario includes, if needed update NIC agent configuration to define properties specific to your environment, such as directory properties.

   Each type of agent has different configuration properties. The output from the **show** command identifies the type of agent under the **agents** hierarchy. For example:

   ```
   VrSaeId {
       configuration {
         directory {
   
   LoginNameVr {
       configuration {
         sae-plug-in {
   ```

For information about agent configuration, see the following sections:

- *Configuring Directory Agents* on page 170
- *Configuring SAE Plug-In Agents* on page 172

Configuring Directory Agents

Use the following configuration statements to configure NIC directory agents:

```
shared nic scenario name agents agent configuration directory {
    search-base search-base;
    search-filter search-filter;
    search-scope (0 | 1 | 2);
    server-url server-url;
    backup-servers-url backup-servers-url;
    principal principal;
    credentials credentials;
}
```

To configure a directory agent:

1. From configuration mode, access the statement that specifies the configuration for the agent.

   ```
   [edit]
   user@host# edit shared nic scenario name agents agent configuration directory
   ```

   For example:

   ```
   [edit]
   user@host# edit shared nic scenario OnePopLogin agents VrSaeId configuration directory
   ```

2. Review the default configuration for the agent. For example:

```
[edit shared nic scenario OnePopLogin agents VrSaeId configuration directory]
user@host# show
search-base o=Network,<base>;
search-filter (objectclass=umcVirtualRouter);
search-scope 2;
server-url ldap://127.0.0.1:389/;
directory-backup-urls ;
principal cn=nic,ou=Components,o=Operators,<base>;
credentials ********;
```

3. (Optional) Change the distinguished name (DN) of the location in the directory from which the agent should read information.

   ```
   [edit shared nic scenario name agents name configuration directory]
   user@host# set search-base search-base
   ```

For example:

```
[edit shared nic scenario OnePop agents PoolVr configuration directory]
user@host# set search-base o=myNetwork,<base>
```

You can use <base> in the DN to refer to the globally configured base DN.

4. (Optional) Change the directory search filter that the agent should use.

   ```
   [edit shared nic scenario name agents name configuration directory]
   user@host# set search-filter search-filter
   ```

   For example:

   ```
   [edit shared nic scenario OnePop agents PoolVr configuration directory]
   user@host# set search-filter objectclass=umcVirtualRouter
   ```

5. (Optional) Change the location in the directory relative to the base DN from which the NIC agent can retrieve information.

   ```
   [edit shared nic scenario name agents name configuration directory]
   user@host# set search-scope (0 | 1 | 2)
   ```

   where:

   - 0—Entry specified in the search-base statement
   - 1—Entry specified in the search-base statement and objects that are subordinate by one level
   - 2—Subtree of entry specified in the search-base statement

6. For an installation on a Solaris platform, specify the location of the directory in URL string format.

   ```
   [edit shared nic scenario name agents name configuration directory]
   user@host# set server-url ldap:// host:portNumber
   ```

For example, to specify the directory on a C-series Controller:

```
[edit shared nic scenario OnePop agents PoolVr configuration directory]
user@host# set server-url ldap://127.0.0.1:389/
```

7. List the URLs of redundant directories. Separate URLs with semicolons.

```
[edit shared nic scenario name agents name configuration directory]
user@host# set directory-backup-urls backup-servers-urls
```

8. Specify the DN that contains the username that the directory server uses to authenticate the NIC agent.

```
[edit shared nic scenario name agents name configuration directory]
user@host# set principal principal
```

For example:

```
[edit shared nic scenario OnePop agents PoolVr configuration directory]
user@host# set principal cn=nic,ou=Components,o=Operators,<base>
```

9. Specify the password that the directory server uses to authenticate the NIC agent.

```
[edit shared nic scenario name agents name configuration directory]
user@host# set credentials credentials
```

10. Restart the NIC agent.

```
user@host>request nic restart agent name name
```

Configuring SAE Plug-In Agents

By default, the CORBA naming server on a C-series Controller uses port 2809. The NIC host is configured to communicate with this naming server; you do not need to change JacORB properties.

Use the following configuration statements to configure NIC SAE plug-in agents:

```
shared nic scenario name agents agent configuration sae-plug-in{
    event-filter event-filter;
    number-of-events number-of-events;
}
```

If you plan to change the event filter for the agent, make sure that you are familiar with:

- Plug-in attributes and values

  See *SRC-PE Subscribers and Subscriptions Guide, Chapter 11, Configuring Accounting and Authentication Plug-Ins with the SRC CLI.*

- Filter syntax

  See the documentation for the SAE CORBA Remote API in the SAE Core API documentation on the Juniper Networks Web site at:

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

To configure an SAE plug-in agent:

1. From configuration mode, access the statement that specifies the configuration for the agent.

   [edit]
   user@host# **edit shared nic scenario** *name* **agents** *agent* **configuration sae-plug-in**

   For example:

   [edit]
   user@host# **edit shared nic scenario OnePopLogin agents LoginNameVr configuration sae plug-in**

2. Review the default configuration for the agent. For example:

```
[edit shared nic scenario OnePopLogin agents LoginNameVr configuration sae-plug-in]
user@host# show
event-filter "(&(!(PA_USER_TYPE=INTF))(!(PA_LOGIN_NAME=[None])))";
number-of-events-sent-in-a-synchronization-call 50;
```

3. (Optional) Change an LDAP filter that change the events that the agent collects.

   [edit shared nic scenario *name* agents *agent* configuration sae-plug-in]
   user@host# **set event-filter** *event-filter*

   Typically, you do not need to change this value. If you do want to filter other events, use the format *pluginAttribute=attributeValue* format for event filters, where:

   - *pluginAttribute*—Plug-in attribute name
   - *attributeValue*—Value of filter

   For example:

   [edit shared nic scenario *name* agents *agent* configuration sae-plug-in]
   user@host# **set event-filter PA_USER_TYPE=INTF**

4. Specify the number of events that the SAE sends to the agent at one time during state synchronization.

   [edit shared nic scenario *name* agents *agent* configuration sae-plug-in]
   user@host# **set number-of-events** *number-of-events*

   For example:

[edit shared nic scenario OnePopLogin agents LoginNameVr configuration sae plug-in]
user@host# **set number-of-events 50**

Configuring the SAE to Communicate with SAE Plug-In Agents When You Use NIC Replication

For each NIC host that uses SAE plug-in agents, configure a corresponding external plug-in for the SAE. By default, the SAE plug-in agents share events with the single SAE plug-in. You must also configure the SAE to communicate with the SAE plug-in agent in each NIC host that you use in the NIC replication.

For information about configuring an external plug-in for the SAE, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 11, Configuring Accounting and Authentication Plug-Ins with the SRC CLI*.

To configure an external plug-in:

1. From configuration mode, access the statement that specifies the configuration for an external plug-in for the SAE that communicates with the agent, and assign the plug-in a unique name.

   ```
   [edit]
   user@host# shared sae configuration plug-ins name name
   ```

2. Configure CORBA object reference for the plug-in.

   ```
   [shared sae configuration plug-ins name name external]
   user@host# corba-object-reference corba-object-reference
   ```

   For the CORBA object reference, use the following syntax:

   host:*port-number*/NameService#*plugInName*

   where:

   - *host*—IP address or name of the machine on which you installed the NIC host that supports the agent

     For local host, use the IP address 127.0.0.1.

   - *port-number*—Port on which the name server runs

     The default port number is 2809.

   - *plugInName*—Name under which the agent is registered in the naming service

     Use the format nicsae_*groupname*/saePort where *groupname* is the name of the replication group. (When replication is not used, the format is nicsae/saePort.)

     For example:

     ```
     [shared sae configuration plug-ins name name external]
     user@host# set corba-object-reference
     corbaname::127.0.0.1:2809/NameService#nicsae/saePort
     ```

3. Configure attributes that are sent to the external plug-in for a NIC host. Because the SAE plug-in agents share the event by default, you configure only one for a NIC host.

   ```
   [shared sae configuration plug-ins name name external]
   user@host# set attr
   [( router-name | user-dn | session-id | user-type | user-ip-address | login-name)]
   ```

   Specify the plug-in options that the agent uses. You must specify the options session-id and router-name, and other options that you specified for the agent's network data types and the agent's event filter. Do not specify attributes options of the PAT_OPAQUE attribute type, such as the option dhcp-packet.

   NOTE: Do not include attributes that are not needed.

4. Reference the NIC as a subscriber tracking plug-in.

   ```
   [edit shared sae group name configuration plugins event-publishers]
   user@host# set subscriber-tracking pool-name
   ```

   For example, for a pool named nic:

   ```
   [edit shared sae group name configuration plugins event-publishers]
   user@host# set subscriber-tracking nic
   ```

Obtaining Interface Configuration Information for OnePopStaticRouteIp

If you use the OnePopStaticRouteIp configuration scenario, you must obtain JUNOS interface configuration information for NIC. To get this information, you must run Network Publisher on a Solaris platform to gather the interface information.

To run Network Publisher on a Solaris platform:

1. Install NIC on a Solaris platform.

   See *SRC-PE Getting Started Guide, Chapter 33, Installing the SRC Software on a Solaris Platform*.

2. On the Solaris platform, edit the */opt/UMC/nic/etc/networkPublisher/config.properties* file and run Network Publisher. When you specify the directory configuration in the file, configure the connection to the directory on a C-series Controller.

   See *Chapter 12, Obtaining Interface Configuration for OnePopStaticRouteIp on Solaris Platforms*.

Related Topics

- *Configuring the NIC with the SRC CLI* on page 163
- *Testing a NIC Resolution with the SRC CLI* on page 176

Configuring Advanced NIC Features

If you want to configure NIC features not available at the basic editing level, set the editing level to advanced or expert and use the CLI Help to obtain information about statement options. For information about setting the CLI editing level, see *SRC-PE CLI User Guide, Chapter 9, Controlling the SRC CLI Environment*.

Verifying Configuration for the NIC with the SRC CLI

After you complete the NIC configuration, verify the local NIC configuration and the NIC configuration scenario information.

To verify NIC configuration:

1. In configuration mode, run the show command at the [edit slot 0 nic] hierarchy level.

   ```
   [edit slot 0 nic]
   user@host# show
   ```

2. In configuration mode, run the show command at the [edit shared nic scenario *name*] hierarchy level.

   For example:

   ```
   [edit shared nic scenario OnePop]
   user@host# show
   ```

Related Topics

- *Configuring the NIC with the SRC CLI* on page 163
- *Testing a NIC Resolution with the SRC CLI* on page 176

Testing a NIC Resolution with the SRC CLI

To test a NIC resolution:

- Run the test nic resolve command.

  ```
  user@host> test nic resolve <locator locator> <key key>
  ```

  where:

  - *locator*—Name of locator that requests information on behalf of an application
  - *key*—Value to be resolved. This value must be of the same NIC data type configured in the NIC locator.

For example:

```
user@host> test nic resolve locator /nicLocators/ip key 10.10.10.10
```

Example: Testing a NIC Resolution

The following example shows a successful resolution for an IP key that has the value 192.168.8.2:

```
user@host> test nic resolve locator /nicLocators/ip key 192.168.8.2
IOR:
000000000000003549444C3A736D67742E6A756E697065722E6E65742F7361652F5365727669
6365416374697661746696F6E456E67696E653A312E3000
0000000000000100000000000000680001020000000000F3137322E32382E3233302E31323000
0022610000000000107372632D382F736165504F412F53
41450000000200000000000000080000000004A414300000000010000001C0000000000010001
0000000105010001000101090000000105010001
user@host>
```

The following example shows an unsuccessful resolution for an IP key that has the value 192.168.8.2:

```
user@host> test nic resolve locator /nicLocators/ip key 192.168.3.2
Failed to resolve key 192.168.3.2 for resolver /nicLocators/ip due to
net.juniper.smgt.gateway.nic.protocol.NICExc
IDL:net/juniper/smgt/gateway/nic/protocol/NICException:1.0
user@host>
```

Related Topics

- *Configuring NIC Test Data with the SRC CLI* on page 205

Stopping a NIC Host on a C-series Controller with the SRC CLI

If you run NIC in client/server mode, you can stop the NIC host independently of the NIC proxy.

To stop a NIC host:

- From operational mode, disable the NIC.

  ```
  user@host> disable component nic
  ```

Related Topics

- *Restarting the NIC with the SRC CLI* on page 178
- *Restarting a NIC Agent with the SRC CLI* on page 178
- *Restarting a NIC Resolver with the SRC CLI* on page 179

Restarting the NIC with the SRC CLI

To restart a NIC host:

- From operational mode, restart the NIC.

```
user@host> request restart nic
```

You can also restart the NIC at the slot level.

Related Topics

- *Stopping a NIC Host on a C-series Controller with the SRC CLI* on page 177
- *Restarting a NIC Agent with the SRC CLI* on page 178
- *Restarting a NIC Resolver with the SRC CLI* on page 179

Restarting a NIC Agent with the SRC CLI

You can restart a NIC agent to have the agent read all data in the directory again. Restart a NIC agent if the agent is not synchronized with the directory, or if you switch from one directory to another.

To restart a NIC agent:

- From operational mode, restart the agent.

```
user@host>request nic restart agent name name
```

You can restart all NIC agents by omitting an agent name for the `request nic restart agent` command.

You can also restart a NIC agent at the slot level.

Related Topics

- *Stopping a NIC Host on a C-series Controller with the SRC CLI* on page 177
- *Restarting the NIC with the SRC CLI* on page 178
- *Restarting a NIC Resolver with the SRC CLI* on page 179

Restarting a NIC Resolver with the SRC CLI

In rare instances, such as when you are troubleshooting a NIC configuration, you may want to restart a NIC resolver.

To restart a NIC resolver:

- From operational mode, restart a resolver.

  ```
  user@host>request nic restart resolver name name
  ```

  You can restart all NIC resolvers by omitting a resolver name for the `request nic restart resolver` command.

You can also restart a NIC resolver at the slot level.

Related Topics

- *Stopping a NIC Host on a C-series Controller with the SRC CLI* on page 177
- *Restarting the NIC with the SRC CLI* on page 178
- *Restarting a NIC Agent with the SRC CLI* on page 178

Changing NIC Configurations with the SRC CLI

If you change the type of NIC resolution that you use in your network (for example, from the OnePop configuration scenario to the OnePopAllRealms configuration scenario), delete any existing data and specify a static DN that identifies the DN for the new NIC configuration scenario; otherwise, the new NIC configuration may not perform resolutions correctly.

To change the type of NIC resolution that you use in your network:

1. Set the editing level for the CLI to expert.

   ```
   user@host> set cli level expert
   ```

2. Disable the NIC:

   ```
   user@host> disable component nic
   ```

3. Delete the NIC configuration data for the existing configuration scenario from the directory.

   ```
   user@host> request nic clear scenario-data
   ```

4. Navigate to the [edit slot 0 nic] hierarchy level.
5. Change the value of the static-dn for the local configuration to identify the location of the DN for the new configuration scenario. For example:

   ```
   [edit slot 0 nic]
   ```

```
user@host# set initial static-dn “l=OnePopSharedIp, l=NIC,
ou=staticConfiguration, ou=Configuration, o=Management,<base>”
```

6. Return to operational mode, and restart the NIC host.

```
user@host>request nic slot number restart
```

7. Set the editing level for the CLI to expert.

```
user@host> set cli level basic
```

8. Configure the new NIC scenario.

Related Topics

- *Configuring the NIC with the SRC CLI* on page 163

Chapter 11

Configuring NIC on a Solaris Platform

This chapter describes how to configure operating parameters for the network information collector (NIC) and manage the NIC on a Solaris platform using the SRC configuration applications that run only on Solaris platforms.

You can also use the SRC CLI to configure operating properties. Use the SRC CLI to configure NIC configuration scenarios. See *Chapter 10, Configuring NIC with the SRC CLI*.

Topics in this chapter include:

- Configuring Operating Parameters for NIC Hosts on a Solaris Platform on page 181
- Starting NIC on a Solaris Platform on page 185
- Stopping a NIC Host on a Solaris Platform on page 186
- Monitoring NIC Hosts on a Solaris Platform on page 186

  See *Chapter 12, Obtaining Interface Configuration for OnePopStaticRouteIp on Solaris Platforms*.

Configuring Operating Parameters for NIC Hosts on a Solaris Platform

The operating parameters define how the NIC host interacts with other SRC components, such as the directory.

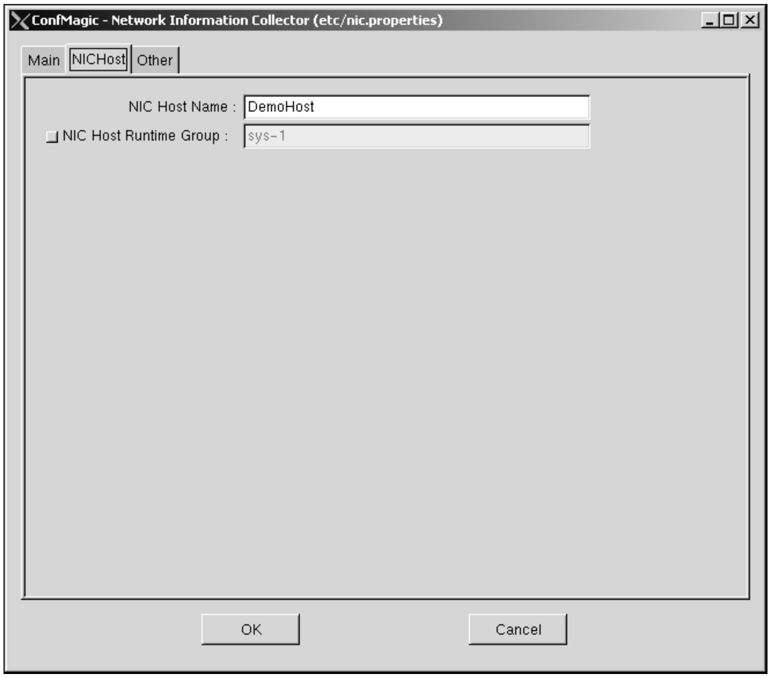
You can also configure NIC operating parameters from the SRC CLI, see *Chapter 10, Configuring NIC with the SRC CLI*.

To configure the operating parameters:

1. Log in as `root`.
2. Start the local configuration tool in the directory where you installed the NIC.

   /opt/UMC/nic/etc/config

   The Network Information Collector window appears.



ConfMagic - Network Information Collector (etc/nic.properties)
Main
NICHost
Other
OK
Cancel

NIC Host Name : DemoHost

☐ NIC Host Runtime Group : sys-1

3. Configure the fields in each tab of this window. The following sections describe the properties on each tab:
   - Directory Connection Properties for NIC Hosts on page 182
   - NIC Host Properties on page 184
   - Additional Properties for NIC Hosts on page 184
4. Click OK.

NOTE: If you change any of the NIC operating parameters, restart NIC for the changes to take effect.

Directory Connection Properties for NIC Hosts

In the Main tab of the Network Information Collector window of the local configuration tool, you can modify the following fields to configure directory connection properties.

Primary Directory Server

- Location of the directory server in URL string format.
- Value—URL in the format [ldap | ldaps]:// { <host> }: <portNumber>
  - ldap—LDAP connection (not secure)
  - ldaps—Secure LDAP connection
  - <host>—Name or IP address of the host that supports the directory
  - <portNumber>—Number of the TCP/IP port

- Default— ldap://127.0.0.1:389/
- Example—ldaps://192.0.2.10:389/

Backup Directory Servers

- List of redundant directories.
- Value—List of URLs separated by semicolons.

  For format of the URL, see the field Primary Directory Server on page 182.
- Example—ldaps://192.0.2.10:389/

Base DN

- Location in the directory in which the SRC data is stored.
- Value—DN
- Example—*o = UMC*

Bind DN

- DN that contains the username that the directory server uses to authenticate the NIC host.
- Value—<DN>, <base>
- Example—*cn=nic, ou=Components, o=Operators, <base>*

Bind Password

- Password that the directory server uses to authenticate the NIC host.
- Value—Text string or Base64 string
- Example—nic

Static Configuration DN

- DN of the location in which the NIC configuration is stored.
- Value—DN
- Example—*l=OnePop, l=NIC, ou=staticConfiguration, ou=Configuration, o=Management, o=umc*

Dynamic Configuration DN

- DN of the location in which data that the NIC automatically generates is stored.
- Value—DN
- Example—*ou=dynamicConfiguration, ou=Configuration, o=Management, o=umc*

Connect Timeout(s)

- Time that the NIC waits for the directory server to respond when it tries to connect to the directory.
- Value—Number of seconds in the range 1–2147483647
- Example—10

NIC Host Properties

In the NIC Host tab of the Network Information Collector window of the local configuration tool, you can modify the following fields.

NIC Host Name

- Name of the NIC host that you configured.
- Value—Text string
- Guidelines—Use the name DemoHost. The configuration scenarios all use DemoHost as the NIC hostname.
- Default—No value
- Example—DemoHost

NIC Host Runtime Group

- Group to which this NIC host belongs if you configure NIC replication.
- Value—Text string
- Default—No value
- Example—ontarioHosts

Additional Properties for NIC Hosts

In the Other tab of the Network Information Collector window of the local configuration tool, you can modify the following fields.

NIC Host Java

- Path to the JRE.
- Value—Path (absolute or relative) to the directory that contains the JRE
- Example—*../jre/bin*

JVM Max Heap

- Maximum memory size available to the JRE.
- Value—Capacity in megabytes

- Guidelines—By default, the JRE can allocate 128 MB. Change this value if you have problems because of lack of memory. Set to a value lower than the available physical memory to avoid low performance because of disk swapping.

  If you use an SAE plug-in agent, we recommend that you increase the JVM max heap to a value in the range 400–500 MB.

- Default—128 MB

Enable Sysman Clients

- Specifies whether or not there is support for viewing SNMP counters with an SNMP browser.
- Value
  - Yes—Enabled
  - No—Disabled
- Default—No

Sysman IOR

- Folder that contains the IOR file for the NIC. The NIC writes its object references to this folder, and the SNMP agent discovers NIC components by monitoring the NIC IOR file in this folder.
- Value—Path to the folder that contains the IOR
- Guidelines—By default, the NIC IOR file is in the *var* folder, which is relative to the SNMP agent installation folder (*/opt/UMC/agent*). You need to change this property only if you installed the SNMP agent in a folder other than the default folder, or if you previously changed this property and now need it to point to the folder where the IOR file currently resides.
- Default—*/opt/UMC/agent/var*

Starting NIC on a Solaris Platform

If you run NIC in client/server mode, after you configure operating parameters for a NIC host and modify basic configuration for a NIC host, start the NIC host.

To start a NIC host:

1. On the machine on which the NIC host is installed, log in as `root` or as an authorized nonroot admin user.
2. Start the NIC host from its installation directory.

   /opt/UMC/nic/etc/nichost start

Stopping a NIC Host on a Solaris Platform

If you run NIC in client/server mode, you can stop the NIC host independently of the NIC proxy.

To stop a NIC host:

1. On the machine on which the NIC host is installed, log in as `root` or as an authorized nonroot admin user.
2. Stop the NIC host from its installation directory.

   ```
   /opt/UMC/nic/etc/nichost stop
   ```

Monitoring NIC Hosts on a Solaris Platform

To verify that a NIC host is running:

1. On the machine on which the NIC host is installed, log in as `root` or as an authorized nonroot admin user.
2. Verify the status of the NIC host from its installation directory.

   ```
   /opt/UMC/nic/etc/nichost status
   ```

Chapter 12

Obtaining Interface Configuration for OnePopStaticRouteIp on Solaris Platforms

This chapter describes how to obtain configuration information for a JUNOS interface for use with the OnePopStaticRouteIp configuration scenario in NIC. Topics include:

JUNOS Interface Information for OnePopStaticRouteIp

The OnePopStaticRouteIp configuration scenario for NIC resolves an IP address for a subscriber whose traffic enters the network through a JUNOS interface to a reference for the SAE that manages the interface. To perform this resolution, the NIC needs information about the JUNOS interfaces. The Threat Mitigation Application Portal relies on the OnePopStaticRouteIp configuration scenario.

The interface information is stored in the directory in an XML document. You can add the interface information to the directory and update the information as needed from the network publisher. The network publisher is a NIC component that collects interface information from the routing tables on specified JUNOS routing platforms. You can view and update the interface configuration from SDX Admin.

Information Collection for OnePopStaticRouteIp from the Network Publisher

The network publisher gathers information about interfaces on specified JUNOS routers and then stores that information in the directory. You run the network publisher whenever you want to get interface information from one or more routers; NIC does not automatically update configuration information in the directory.

The network publisher uses a configuration file on the system on which the NIC host is configured. When you run the network publisher, the information in the configuration file determines the types of information to be collected.

The network publisher is supported on Solaris platforms that have the SRC software installed.

Before You Run the Network Publisher

When you run the network publisher, it connects to a number of JUNOS routing platforms through Telnet.

Before you run the network publisher:

- Verify the version of the JUNOS software that is running on each JUNOS routing platform.

  Typically, all of the JUNOS routing platforms should run the same version of the JUNOS software.

- Make sure that a Telnet service is enabled on each router from which the network publisher is to collect interface information.

Configuring the Network Publisher

To configure the network publisher:

- In a text editor, edit the file */opt/UMC/nic/etc/networkPublisher/config.properties.*

  For information about the fields in this file, see *Network Publisher Configuration File Fields* on page 189.

Network Publisher Configuration File Fields

The */opt/UMC/nic/etc/networkPublisher/config.properties* file contains the following types of configuration fields:

- *Logging Configuration Fields* on page 189
- *Router Configuration Fields* on page 189
- *Filter Configuration Fields* on page 190
- *Directory Configuration Fields* on page 191
- *Troubleshooting Configuration Fields* on page 192

Entries in the file have the format <field> = <value>.

The network publisher identifies routers by a number; for example, r1, r2.

Logging Configuration Fields

The network publisher uses the same logging properties as other SRC components.

For information about logging properties and about managing log files, see:

- *SRC-PE Monitoring and Troubleshooting Guide, Chapter 5, Managing SRC Log Files on a Solaris Platform*

Router Configuration Fields

The router configuration fields provide information about the JUNOS routing platforms on which interfaces reside for which the network publisher collects information.

/routers/tags.release

- Release number of the JUNOS software installed on the JUNOS routing platform.
- Value—Release number in the format:

  <major release>.<minor release>R<revision>
- Guidelines—If a value is also specified under */routers/r<number>tags.release*, the value for the specified router is used for that router.
- Default—No value
- Example—/routers/tags.release = 7.6R1

/routers/tags.hostname

- Hostname of the machine on which the network publisher runs.
- Value—<hostname>
- Default—No value
- Example—/routers/tags.hostname = myhost

/routers/junoscript-authentication.username

- Username to log in to the JUNOScript server.
- Value—<username>
- Default—No value
- Example—/routers/junoscript-authentication.username = root

/routers/junoscript-authentication.challenge_response

- Password to log in to the JUNOScript server
- Value—<password>
- Default—No value
- Example—/routers/junoscript-authentication.challenge_response = secret

/routers/r<number>/hostname

- Hostname of a JUNOS routing platform.
- Value—<hostname>
- Default—No value
- Example—/routers/r1/hostname = RouterExternal

/routers/r<number>/address

- IP address of the JUNOS routing platform for which you specified a hostname.
- Value—<IP address in dotted decimal notation>
- Default—No value
- Example—/routers/r1/address = 10.10.10.10

/routers/r<number>/tags.release

- Release number of the JUNOS software installed on a specific JUNOS routing platform.
- Value—Release number in the format:

  <major release>.<minor release>R<release number>
- Guidelines—If a value is also specified under */routers/tags.release*, the value for the specified router is used for that router.
- Default—No value
- Example—/routers/r1/tags.release = 7.6R1

Filter Configuration Fields

The filter configuration fields specify filters that the network publisher uses to collect information from JUNOS routing platforms. You can specify two filters.

/transform/route_table_filter

- Routing table from which the network publisher collects interface information.
- Value—Element name in the format:

  (element-name = <value>)
- Default—(table-name = inet.0)
- Example—/transform/route_table_filter = (table-name = inet.0)

/transform/route_entry_filter

- Element(s) in a specified router table from which the network publisher collects interface information.
- Value—Element name in the format:

  (<element-name> = <value>)
- Default—(protocol-name = *)
- Example—/transform/route_entry_filter = (protocol-name = *)

Directory Configuration Fields

The directory fields specify information used to connect to the directory.

/dir/java.naming.provider.url

- URL of the primary directory.
- Value—URL in the format ldap:// <host>:389

  <host>—IP address or name of directory host
- Default—No value
- Example—/dir/java.naming.provider.url = ldap://127.0.0.1:389/

/dir/java.naming.security.principal

- Distinguished name (DN) of the directory entry that defines the username with which the network publisher accesses the directory.
- Value—<DN>
- Default—No value
- Example—/dir/java.naming.security.principal = cn=umcadmin, o=umc

/dir/java.naming.security.credentials

- Password with which the network publisher accesses the directory.
- 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—No value

- Example
  - /dir/java.naming.security.credentials = admin123
  - /dir/java.naming.security.credentials = {BASE64}c3Nw

/dir/baseDN

- Subtree in the directory that stores router data.
- Value—<DN>
- Default—*o=Network, o=umc*
- Example—/dir/baseDN=o=Network, o=umc

Troubleshooting Configuration Fields

The troubleshooting configuration fields let you specify file information that you can use to troubleshoot configuration for the network publisher.

/routers/r<number>/session_type

- File that contains properties for the routers.
- Value—<filename>
- Guidelines—By default, the network publisher obtains router information through a Telnet session. You can specify an input file for one or more routers to troubleshoot the configuration for the network publisher. The values in an input file for a specified router take precedence over values obtained from the router through a Telnet session.
- Default—LocalFile
- Example—/routers/r1/session_type = LocalFile

/routers/r<number>/input_dir

- Directory that contains the <router_name>_1.xml document (where router_name is the hostname of a router), which contains router properties for the network publisher.
- Value—<directory-name>
- Guidelines—Use a file in the input directory if you do not want to connect to the router to obtain the interface configuration information. Use a file defined by routers/r<number>/session_type in this directory to troubleshoot the configuration for the network publisher.
- Default—/opt/UMC/nic/sample/junos/rt
- Example—/routers/r1/input_dir =/opt/UMC/nic/myconfig

/routers/r<number>/output_dir

- Directory that contains the <router_name>_1.xml document (where router_name is the hostname of a router), which contains interface configuration information collected from the routing table on a JUNOS routing platform.
- Value—<directory-name>

- Guidelines—You must specify an output directory for information to be written to an output file. You can read the information stored in files in this directory to determine whether they contain the expected information from the routing table on the specified JUNOS routing platform.
- Default—opt/UMC/nic/var/junos/rt
- Example—/routers/r1/output_dir = /var/junos/mydir

Running the Network Publisher

You run the network publisher on a Solaris platform each time you want to collect information about interfaces on JUNOS routing platforms.

To run the network publisher:

1. Move to the *etc* directory that is under the NIC installation directory, typically */opt/UMC/nic/etc*.
2. Run the **networkPublisher appl** command:

   ```
   ./networkPublisher appl
   ```

Troubleshooting Router Connections and Configuration for the Network Publisher

You can troubleshoot the connection between the network publisher and one or more routers, and the configuration on the routers by providing configuration information to the publisher from a file rather than from JUNOS routing platforms.

To specify that the network publisher obtain router configuration information from a file:

1. Edit the */opt/UMC/nic/etc/networkPublisher/config.properties* file.
2. Specify an input directory for the routers by configuring the following property in the file:

   ```
   /routers/r<number>/input_dir =<directory>
   ```

   For example:

   ```
   /routers/r1/input_dir =/opt/UMC/nic/myconfig
   ```

3. Specify that the network publisher obtain properties for a router from a file on the local system, instead of from the directory, by configuring the following property in the file:

   ```
   /routers/r<number>/session_type = LocalFile
   ```

   For example:

   ```
   /routers/r1/session_type=LocalFile
   ```

By default, the input file is */opt/UMC/nic/sample/junos/rt/<router_name>_1.xml* where router_name is the hostname of a JUNOS routing platform.

You can change the location of the input directory.

See *Changing the Location of an Input Directory for the Network Publisher* on page 194.

Changing the Location of an Input Directory for the Network Publisher

By default, an input file is located in the directory */opt/UMC/nic/sample/junos/rt*. You can use an input file to troubleshoot the configuration of the network publisher.

You can define a different location for this file in the network publisher configuration in the */opt/UMC/nic/etc/networkPublisher/config.properties* file.

To change the location of an input file that contains router configuration:

- Edit the *config.properties* file. Specify that the network publisher locate an input file by configuring the following property in the file:

  ```
  /routers/r<number>/input_dir =
  ```

  See *Network Publisher Configuration File Fields* on page 189.

Reviewing the Information Collected from a JUNOS Routing Platform

To review information that the network publisher collects from a JUNOS routing platform:

1. Locate the directory that contains the output file(s) by reviewing the value of the following properties in the */opt/UMC/nic/etc/networkPublisher/config.properties* configuration file:
   - For one router:

     ```
     /routers/r1/output_dir =
     ```

   - For all routers:

     ```
     /routers/output_dir =
     ```

2. In the directory specified `/routers/r1/output_dir`, open the *<router_name>_1.xml* document (where router_name is the hostname of a JUNOS routing platform), and review the file content.

If the information in the file is different from the information expected, there may be a problem with the configuration on the router.

Reviewing and Editing Interface Information from SDX Admin

You can use SDX Admin to edit the XML document that contains interface information used by the OnePopStaticRouteIp configuration scenario for NIC. You use the network publisher to collect the interface information and populate the XML document in the directory for you.

To use SDX Admin to edit the XML document that provides interface information for JUNOS routing platforms:

1. In the navigation pane, select a router under *o = network, o = umc*.
2. Click the **Interface Configuration** tab, and edit values in the content pane.

   For information about file syntax and a sample file, see *NIC Document That Maps Subscriber IP Addresses to a JUNOS Interface* on page 195.

NIC Document That Maps Subscriber IP Addresses to a JUNOS Interface

NIC stores information about subscriber IP addresses that map to JUNOS interfaces on which the associated traffic enters the network. This XML document is stored in the specified directory. These files comply with the syntax in the file */opt/UMC/nic/etc/networkConfig.xsd*. An example file */opt/UMC/nic/networkConfig.xml* shows the type of information generated by the network publisher.

For an example configuration in SDX Admin, see the Interface Configuration tab for *o = OrderedCimKeys = THMA1, o = network, o = umc*.

Chapter 13

Configuring Applications to Communicate with an SAE

This chapter provides information about configuring NIC proxies. Topics include:

- Overview of NIC Proxy Configuration on page 197
- Before You Configure a NIC Proxy on page 197

Overview of NIC Proxy Configuration

You configure applications to communicate with network information collector (NIC) hosts. A NIC host can be local within an application, or external to the application. For Java applications, you also configure NIC proxies as part of an application.

For a number of SRC components, such as the SRC Volume-Tracking Application (SRC-VTA) and the Dynamic Service Activator, you can configure the NIC proxy for the application from the SRC CLI. For other applications, such as the sample residential portal, you configure the NIC proxy in a property file. If you configure a NIC proxy from a property file, the fields are the same as the fields that appear at the CLI. When you develop and test SRC components that use a NIC, you can configure a NIC proxy stub to take the place of the NIC host.

For more information about NIC proxies, see *Chapter 9, Locating Subscriber Information with the NIC*.

Before You Configure a NIC Proxy

The values that you configure for a NIC proxy depend on the particular application; for example, you must specify the type of data used for the key and the type of data used for the value for each application.

Before you configure a NIC proxy for an application, obtain the following information from the system manager who maintains the NIC configuration for NIC hosts:

- The name of the resolver that the application uses.
- The type of key the application will provide to the NIC host.
- The type of value the NIC host is to return.
- Whether or not the application will use a local NIC host.
- If the application does not use a local NIC host:
    - The size of the NIC proxy cache.
    - The groups to be listed for NIC host selection. These groups provide NIC replication.

Also, if you use the SRC software on a Solaris platform and use a Java Runtime Environment (JRE) other than the one included in the SRC software distribution, review the configuration for the object request broker (ORB) to ensure that it meets the requirements for the NIC.

For information about the ORB, see *Chapter 15, Configuring Applications to Communicate with an SAE with SDX Admin.*

Chapter 14

Configuring SRC Applications to Communicate with an SAE with the SRC CLI

You can use the CLI to configure SRC applications to communicate with network information collector (NIC) hosts. This chapter describes how to configure a NIC proxy from the SRC CLI on a C-series Controller or on a Solaris platform running the SRC software. Topics include:

- Configuration Statements for NIC Proxies on page 199
- Before You Configure a NIC Proxy on page 200
- Configuring Resolution Information for a NIC Proxy with the SRC CLI on page 201
- Changing the Configuration for the NIC Proxy Cache with the SRC CLI on page 202
- Configuring a NIC Proxy for NIC Replication with the SRC CLI on page 203
- *Configuring NIC Test Data with the SRC CLI* on page 205

Configuration Statements for NIC Proxies

Use the following configuration statements to configure a NIC proxy for the SAE from the [edit] hierarchy level.

```
shared sae configuration nic-proxy-configuration name {
}

shared sae configuration nic-proxy-configuration name resolution {
    resolver-name resolver-name;
    key-type key-type;
    value-type value-type;
    expect-multiple-values;
    constraints constraints;
}
```

```
shared sae configuration nic-proxy-configuration name cache {
      cache-size cache-size;
      cache-cleanup-interval cache-cleanup-interval;
      cache-entry-age cache-entry-age;
}

shared sae configuration nic-proxy-configuration name nic-host-selection {
      groups groups;
      selection-criteria (roundRobin | randomPick | priorityList);
}

shared sae configuration nic-proxy-configuration name nic-host-selection blacklisting {
      try-next-system-on-error;
      number-of-retries-before-blacklisting number-of-retries-before-blacklisting;
      blacklist-retry-interval blacklist-retry-interval;
}
```

To configure a NIC proxy stub for SAE, use the following statements:

```
shared sae configuration nic-proxy-configuration name test-nic-bindings {
      use-test-bindings;
}

shared sae configuration nic-proxy-configuration name test-nic-bindings key-values
name {
      value;
}
```

Before You Configure a NIC Proxy

Before you configure a NIC proxy, you should have a good understanding of:

- NIC resolution
- NIC data types
- How NIC proxies work

See *Chapter 9, Locating Subscriber Information with the NIC*; *Chapter 17, NIC Resolution Process*; and *Chapter 13, Configuring Applications to Communicate with an SAE*.

NOTE: You cannot configure a local NIC host when the NIC is running on a C-series Controller.

Configuring Resolution Information for a NIC Proxy with the SRC CLI

Use the following configuration statements to configure a NIC proxy:

```
shared sae configuration nic-proxy-configuration name {
}

shared sae configuration nic-proxy-configuration name resolution {
    resolver-name resolver-name;
    key-type key-type;
    value-type value-type;
    expect-multiple-values;
    constraints constraints;
}
```

To configure resolution information for a NIC proxy.

1. From configuration mode, access the configuration statement that specifies the NIC proxy configuration.

   ```
   [edit]
   user@host# edit shared sae configuration nic-proxy-configuration name
   resolution
   ```

   For example:

   ```
   [edit]
   user@host# edit shared sae configuration nic-proxy-configuration ip resolution
   ```

2. Specify the NIC resolver that this NIC proxy uses.

   ```
   [edit shared sae configuration nic-proxy-configuration ip resolution]
   user@host# set resolver-name resolver-name
   ```

   This resolver must be the same as one that is configured on the NIC host. For example:

   ```
   [edit shared sae configuration nic-proxy-configuration ip resolution]
   user@host# set resolver-name /realms/ip/A1
   ```

3. Specify the NIC data type that the key provides for the NIC resolution.

   ```
   [edit shared sae configuration nic-proxy-configuration ip resolution]
   user@host# set key-type key-type
   ```

   For example:

   ```
   [edit shared sae configuration nic-proxy-configuration ip resolution]
   user@host# set key-type ip
   ```

   To qualify data types, enter a qualifier within parentheses after the data type; for example, to specify username as a qualifier for the key LoginName:

   ```
   [edit shared sae configuration nic-proxy-configuration ip resolution]
   user@host# set key-type LoginName (username)
   ```

4. Specify the type of value to be returned in the resolution for the application that uses the NIC proxy.

   ```
   [edit shared sae configuration nic-proxy-configuration ip resolution]
   user@host# set value-type value-type
   ```

   For example:

   ```
   [edit shared sae configuration nic-proxy-configuration ip resolution]
   user@host# set value-type SaeId
   ```

5. (Optional) If the key can have more than one value, specify that the key can have multiple corresponding values.

   ```
   [edit shared sae configuration nic-proxy-configuration ip resolution]
   user@host# set expect-multiple-values
   ```

6. (Optional. Available at the Advanced editing level.) If the application provides a constraint in the resolution request, specify the data type for the constraint. The constraint represents a condition that must or may be satisfied before the next stage of the resolution process can proceed.

   ```
   [edit shared sae configuration nic-proxy-configuration ip resolution]
   user@host# set constraints constraints
   ```

Changing the Configuration for the NIC Proxy Cache with the SRC CLI

You can modify cache properties for the NIC proxy to optimize the resolution performance for your network configuration and system resources. Typically, you can use the default settings for the cache properties. The configuration statements are available at the Advanced editing level.

Use the following configuration statements to change values for the NIC proxy cache:

```
shared sae configuration nic-proxy-configuration name cache {
    cache-size cache-size;
    cache-cleanup-interval cache-cleanup-interval;
    cache-entry-age cache-entry-age;
}
```

To configure the cache for a NIC proxy:

1. From configuration mode, access the configuration statement that specifies the NIC proxy configuration. For example:

   ```
   [edit]
   user@host# edit shared sae configuration nic-proxy-configuration ip cache
   ```

2. Specify the maximum number of keys for which the NIC proxy retains data.

   ```
   [edit shared sae configuration nic-proxy-configuration ip cache]
   user@host# set cache-size cache-size
   ```

   If you decrease the cache size or disable the cache while the NIC proxy is running, the NIC proxy removes entries in order of descending age until the cache size meets the new limit.

3. Specify the time interval at which the NIC proxy removes expired entries from its cache.

   ```
   [edit shared sae configuration nic-proxy-configuration ip cache]
   user@host# set cache-cleanup-interval cache-cleanup-interval
   ```

4. Specify the how long an entry remains in the cache.

   ```
   [edit shared sae configuration nic-proxy-configuration ip cache]
   user@host# set cache-entry-age cache-entry-age
   ```

Configuring a NIC Proxy for NIC Replication with the SRC CLI

Typically, you configure NIC replication to keep the NIC highly available. You configure NIC host selection to specify the groups of NIC hosts to be contacted to resolve a request, and to define how the NIC proxy handles NIC hosts that the proxy is unable to contact. The configuration statements are available at the Advanced editing level.

Use the following configuration statements to configure NIC host selection for a NIC proxy:

```
shared sae configuration nic-proxy-configuration name nic-host-selection {
    groups groups;
    selection-criteria (roundRobin | randomPick | priorityList);
}

shared sae configuration nic-proxy-configuration name nic-host-selection blacklisting {
    try-next-system-on-error;
    number-of-retries-before-blacklisting number-of-retries-before-blacklisting;
    blacklist-retry-interval blacklist-retry-interval;
}
```

To configure a NIC proxy to use NIC replication:

1. From configuration mode, access the configuration statement that specifies the NIC proxy configuration. For example:

   ```
   [edit]
   user@host# edit shared sae configuration nic-proxy-configuration ip
   ```

2. Specify the list of groups of NIC hosts that the NIC proxy can contact for resolution requests. Use commas to separate the group names.

   ```
   [edit shared sae configuration nic-proxy-configuration ip nic-host-selection]
   user@host# set groups groups
   ```

   For example

   ```
   [edit shared sae configuration nic-proxy-configuration ip nic-host-selection]
   user@host# set groups [group1 group2]
   ```

3. If you configure more than one group, specify the selection criteria that the NIC proxy uses to determine which NIC host to contact.

   ```
   [edit shared sae configuration nic-proxy-configuration ip nic-host-selection]
   user@host# set selection-criteria (roundRobin | randomPick | priorityList)
   ```

   where:

   - roundRobin—NIC proxy selects NIC hosts in a fixed, cyclic order. The NIC proxy always selects the next host in the list.
   - randomPick—NIC proxy selects NIC hosts randomly from the list.
   - priorityList—NIC proxy selects NIC hosts according to their assigned priorities in the list. If the host with the highest priority in the list is not available, the NIC proxy tries the host with the next-highest priority, and so on.

     Priorities are defined by the order in which you specify the groups. You can change the order of NIC hosts in the list by using the insert command.

4. Access the configuration statement that specifies the NIC proxy configuration for blacklisting—the process of handling nonresponsive NIC hosts.

   ```
   [edit shared sae configuration nic-proxy-configuration ip nic-host-selection]
   user@host# edit blacklisting
   [edit shared sae configuration nic-proxy-configuration ip nic-host-selection
   blacklisting]
   ```

5. Specify whether or not the NIC proxy should contact the next specified NIC host if a NIC host is determined to be unavailable.

   ```
   [edit shared sae configuration nic-proxy-configuration ip nic-host-selection
   blacklisting]
   user@host# set try-next-system-on-error
   ```

6. (Optional) Change the number of times the NIC proxy tries to communicate with a NIC host before the NIC proxy stops communicating with the NIC host for a period of time. The default is 3.

   ```
   [edit shared sae configuration nic-proxy-configuration ip nic-host-selection
   blacklisting]
   user@host# set number-of-retries-before-blacklisting
   number-of-retries-before-blacklisting
   ```

7. (Optional) Change the interval at which the NIC proxy attempts to connect to an unavailable NIC host. The default is 15 seconds.

   ```
   [edit shared sae configuration nic-proxy-configuration name nic-host-selection
   blacklisting]
   user@host# set blacklist-retry-interval blacklist-retry-interval
   ```

Configuring NIC Test Data with the SRC CLI

To test a resolution without NIC, you can configure a NIC proxy stub to take the place of the NIC. The NIC proxy stub comprises a set of explicit mappings of data keys and values in the NIC proxy configuration. When the SAE (or another SRC component configured to use a NIC proxy stub) passes a specified key to the NIC proxy stub, the NIC proxy stub returns the corresponding value. When you use a NIC proxy stub, no NIC infrastructure is required.

For example, you can specify a subscriber’s IP address that is associated with a particular SAE. When the SRC component passes this IP address to the NIC proxy stub, the NIC proxy stub returns the corresponding SAE.

To use the NIC proxy stub for the SAE:

1. In configuration mode, navigate to the NIC proxy configuration and specify the type of key you want to map to a value.

   ```
   [edit shared sae configuration nic-proxy-configuration name]
   user@host# set resolution key-type key-type
   ```

   For example, to specify the key ip for the ip NIC proxy configuration:

   ```
   [edit shared sae configuration nic-proxy-configuration ip]
   user@host# set resolution key-type ip
   ```

2. Enable a NIC proxy stub for a resolution.

   ```
   [edit shared sae configuration nic-proxy-configuration ip]
   user@host# set test-nic-bindings user-test-bindings
   ```

3. Specify the values of the keys for testing. These statements are available at the Expert CLI editing level.

   ```
   [edit shared sae configuration nic-proxy-configuration ip]
   user@host# set test-nic-bindings key-values name value
   ```

   where:

   - *name*—Indicates the NIC data value for the proxy.
   - *value*—Specifies a value for the NIC data type.

For example, to set up a login name to IP mapping for login name jane@virneo.com to the IP address 192.0.2.30:

```
[edit shared sae configuration nic-proxy-configuration ip]
user@host# set test-nic-bindings key-values jane@virneo.com 192.0.2.30
```

For example, to set up an IP to SAE ID mapping for IP address 190.0.2.30 to SAE ID identified by the URL for the CORBA IOR corbaloc::10.227.7.145:8801/SAE:

```
[edit shared sae configuration nic-proxy-configuration ip]
user@host# set test-nic-bindings key-values 192.0.2.30
corbaloc::10.20.7.145:8801/SAE
```

NOTE: The SAE writes the value of the CORBA IOR to the *var/run* directory. The IP address in the corbaloc URL can be adjusted to the IP address or DNS name of the SAE.

You can use the key ANY-KEY to match any key for any key type. For example, if you want all IP addresses to resolve to the same SAE:

```
[edit shared sae configuration nic-proxy-configuration ip]
user@host# set test-nic-bindings key-values ANY-KEY
corbaloc::10.20.7.145:8801/SAE
```

Chapter 15

Configuring Applications to Communicate with an SAE with SDX Admin

This chapter discusses how to ensure that a Solaris platform is configured to support NIC proxies, how to test NIC applications in a Solaris platform, and how to monitor NIC on Solaris platforms. Topics Include:

- Reviewing and Updating the ORB Configuration for Applications That Include a NIC Proxy on Solaris on page 207
- Testing Applications by Using a NIC Proxy Stub on Solaris Platforms on page 210
- Monitoring NIC Proxies on Solaris Platforms on page 213

Reviewing and Updating the ORB Configuration for Applications That Include a NIC Proxy on Solaris

The JRE package (UMCjre) included in the SRC software distribution is preconfigured with JacORB.

JacORB meets the requirements for applications that include a NIC proxy. If you use a different JRE, you must ensure that it is configured with an ORB that supports value types with the Object Management Group's CORBA 2.6 standard.

If the default Java Virtual Machine (JVM) for the Web application server is UMCjre or another environment that complies with this standard, you do not need to configure the ORB. However, if this is not the case, you must configure the ORB to enable your application to communicate with the NIC.

Depending on the type of application, you can do one of the following:

- Configuring JacORB as the Default ORB on page 208
- Configuring One Web Application to Use JacORB on page 209
- Configuring a Web Application Server to Use JacORB on page 209

  In this case, all Web applications, but not other Java applications, inside the Web application server will use this ORB.

For additional information about JacORB, see:

http://www.jacorb.org/documentation.html

For information about the Object Management Group's CORBA 2.6 standard:

http://www.omg.org

For information about how to set up the configurations for ORBs other than JacORB, see the documentation for that ORB.

For information about installing this package, see *SRC-PE Getting Started Guide, Chapter 33, Installing the SRC Software on a Solaris Platform*.

Configuring JacORB as the Default ORB

To configure JacORB as the default ORB for the JRE:

1. Access the folder in the SRC software distribution that contains the files you require for the version of JRE that you are using.
   - For JRE 1.3, access the folder *SDK/lib-1.3*.

     ```
     cd /cdrom/cdrom0/SDK/lib-1.3
     ```

   - For JRE 1.4 or greater, access the folder *SDK/lib-1.4*.

     ```
     cd /cdrom/cdrom0/SDK/lib-1.4
     ```

2. Copy the property files from the folder in the SRC software distribution to the folder *jre/lib* in your JRE installation.

   ```
   cp jacorb.properties <jreInstallDirectory>/jre/lib/jacorb.properties
   cp orb.properties <jreInstallDirectory>/jre/lib/orb.properties
   ```

3. Copy the appropriate JAR files from the folder in the SRC software distribution to the directory *jre/lib/ext* in your JRE installation.
   - For JRE 1.3, copy the file *jacorb.jar*.

     ```
     cp jacorb.jar <jreInstallDirectory>/jre/lib/ext/jacorb.jar
     ```

   - For JRE 1.4 or greater, copy the file *jacorb.jar*.

     ```
     cp jacorb.jar <jreInstallDirectory>/jre/lib/ext/jacorb.jar
     ```

Configuring One Web Application to Use JacORB

To configure a particular Web application that includes the NIC proxy to use JacORB:

1. Access the folder in the SRC software distribution that contains the files you require for the version of JRE that the Web application server is using.
   - For JRE 1.3, access the folder *SDK/lib-1.3*.

     ```
     cd /cdrom/cdrom0/SDK/lib-1.3
     ```

   - For JRE 1.4, access the folder *SDK/lib-1.4*.

     ```
     cd /cdrom/cdrom0/SDK/lib-1.4
     ```

2. Copy the appropriate files from the folder in the SRC software distribution to the folder *WEB-INF/lib* of the Web application.
   - For JRE 1.3, copy the files *jacorb.properties* and *jacorb.jar*.

     ```
     cp jacorb.properties <webAppDirectory>/WEB-INF/lib/jacorb.properties
     cp jacorb.jar <webAppDirectory>/WEB-INF/lib/jacorb.jar
     ```

   - For JRE 1.4, copy the files *jacorb.properties* and *jacorb.jar*.

     ```
     cp jacorb.properties <webAppDirectory>/WEB-INF/lib/jacorb.properties
     cp jacorb.jar <webAppDirectory>/WEB-INF/lib/jacorb.jar
     ```

3. Configure the NIC factory used by the Web application to use this ORB.

   See *Chapter 16, Developing Applications That Use NIC*.

Configuring a Web Application Server to Use JacORB

To configure all Web applications, but not other Java applications, to use JacORB:

1. Access the folder in the SRC software distribution that contains the files you require for the version of JRE that the Web application server is using.
   - For JRE 1.3, access the folder *SDK/lib-1.3*.

     ```
     cd /cdrom/cdrom0/SDK/lib-1.3
     ```

   - For JRE 1.4, access the folder *SDK/lib-1.4*.

     ```
     cd /cdrom/cdrom0/SDK/lib-1.4
     ```

2. For JRE 1.3 and JRE 1.4, include the file *jacorb.jar* file in the classpath for the Web application server.
3. Include the file *jacorb.properties* for the appropriate JRE release in a directory specified in classpath, in the current directory, or in the home directory of the user who starts the Web application server.

4. Configure JacORB to be the ORB for the Web application server ORB. For information about this step, see the JacORB documentation at

   http://www.jacorb.org/documentation.html

Testing Applications by Using a NIC Proxy Stub on Solaris Platforms

To test an application without NIC, you can configure a NIC proxy stub to take the place of the NIC. The NIC proxy stub comprises a set of explicit mappings of data keys and values in the namespace that contains the NIC proxy properties. When the SRC component passes a specified key to the NIC proxy stub, the NIC proxy stub returns the corresponding value.

For example, you can specify a subscriber's IP address that is associated with a particular SAE. When the SRC component passes this IP address to the NIC proxy stub, the NIC proxy stub returns the corresponding SAE.

Configuring a NIC Proxy Stub from SDX Admin

To use SDX Admin to configure a NIC proxy stub:

1. In the navigation pane, select the entry for the NIC proxy.
2. Add the following line to the NIC proxy properties.

Gateway.nic.NicProxyClassName = net.juniper.smgt.gateway.gal.proxy.NicProxyStub

For example, for Dynamic Service activator, located under *l = DynamicServiceActivation, l = WebApplication, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*, you would add the lines similar to the following:

```
/nicProxies/ip/Gateway.nic.NicProxyClassName =
net.juniper.smgt.gateway.gal.proxy.NicProxyStub
/nicProxies/ip/ANY_KEY = corbaloc::192.2.7.100:8801/SAE
```

When you use a NIC proxy stub, you must also configure test data for the stub to use.

See *Configuring the Test Data* on page 210.

Configuring the Test Data

To use a NIC proxy stub, you configure test data for the NIC proxy to use. You can specify that the test data indicate that any key return a specific SAE or that one or more keys map to particular values. If you specify an explicit SAE for a key, the NIC proxy stub returns the IOR for that SAE, rather than the value defined for the ANY_KEY property.

To configure test data, do one of the following:

- Configure a NIC proxy stub to use a corbaloc URL.

  See *Configuring a NIC Proxy Stub to Use a corbaloc URL to Test Data* on page 211.

- Configure a NIC proxy stub to use a file URL.

  See *Configuring a NIC Proxy Stub to Use a File URL to Test Data* on page 211.

- Configure a NIC proxy stub to use an IOR.

  See *Configuring a NIC Proxy Stub to Use an IOR to Test Data* on page 212.

Configuring a NIC Proxy Stub to Use a corbaloc URL to Test Data

To configure a NIC proxy stub to use the corbaloc URL:

1. In the NIC proxy configuration, add a line in the format
   corbaloc::<host>:<port>/SAE

   - <host>—Name or IP address of the SAE.

   - <portNumber>—TCP/IP port number for the SAE. The default is 8801.

   For example, corbaloc::127.0.0.1.145:8801/SAE.

2. In the NIC proxy configuration, add a line to return any key to a specific SAE or a key that the NIC proxy receives.

   To return any key, add a line in the format
   ANY_KEY=corbaloc::<host>:<port>/SAE

   For example, ANY_KEY=corbaloc::sae1:8801/SAE

   To specify explicit mapping between keys and values, add lines in the following format to the NIC proxy configuration.

   <mapping> =corbaloc::<host>:<port>/SAE

   For example, the following test data comprises two subscriber IP addresses associated with different SAEs. You define two explicit mappings:

   192.0.2.10 = corbaloc::sae1:8801/SAE
   192.0.2.11 = corbaloc::sae2:8801/SAE

Configuring a NIC Proxy Stub to Use a File URL to Test Data

To configure a NIC proxy stub to use the IOR file:

1. In the NIC proxy configuration, add a line in the format file://<absolute path to the IOR file.

   For example, file://*/opt/UMC/sae/var/run/sae.ior*

2. In the NIC proxy configuration, add a line to return any key to a specific SAE or a key that the NIC proxy receives.

   To return any key, add a line in the format ANY_KEY = file://<absolute path to the IOR file>.

   For example, ANY_KEY = file://*/opt/UMC/sae/var/run/sae.ior*

   To specify explicit mapping between keys and values, add lines in the following format to the NIC proxy configuration.

   <mapping> = file://<absolute path to the IOR file>

   For example, the following test data comprises two subscriber IP addresses associated with the same SAE. You define two explicit mappings:

   192.0.2.0 = file://*/opt/UMC/sae/var/run/sae.ior*
   192.0.2.1 = file://*/opt/UMC/sae/var/run/sae.ior*

Configuring a NIC Proxy Stub to Use an IOR to Test Data

To configure a NIC proxy stub to use a copy of the IOR:

1. Access the *sae.ior* file in the directory */opt/UMC/sae/var/run*.

2. Copy the complete IOR of the SAE from this file.

3. In the NIC proxy configuration, add a line to return any key to a specific SAE or a key that the NIC proxy receives.

   To return any key, add a line in the format ANY_KEY = <SAE_IOR>.

   - <SAE_IOR>—IOR that you copied

   For example:

```
ANY_KEY =
IOR:000000000000003549444C3A736D67742E6A756E697065722E6E65742
F7361652F5365727669636541637469766174696F6E456E67696E653A312
E30000000000000000200000000000000070000102000000000D31302E323
2372E312E323031000022610000001B5374616E64617264496D706C4E616
D652F736165504F412F534145000000000200000000000000080000000004
A414300000000010000001C0000000000010001000000010501000100010
1090000000105010001000000010000002C0000000000000001000000010
000001C00000000000100010000000105010001000101090000000105010
001
```

   To specify explicit mapping between keys and values, add lines in the following format to the NIC proxy configuration.

   <key> = <value>

   For example, the following test data comprises two subscriber IP addresses associated with different SAEs. You can define two explicit mappings:

```
192.0.2.0 =
IOR:000000000000003549444C3A736D67742E6A756E697065722E6E65742
F7361652F5365727669636541637469766174696F6E456E67696E653A312
E300000000000000002000000000000007000010200000000000D31302E323
2372E312E323031000022610000001B5374616E64617264496D706C4E616
D652F736165504F412F53414500000000020000000000000000800000000004
A414300000000010000001C0000000000010001000000010501000100010
1090000000105010001000000010000002C0000000000000001000000010
000001C00000000000100010000000105010001000101090000000105010
001
192.0.2.1 =
IOR:000000000000002438444C3A736D67742E6A756E697065722E6E65742
F7361652F5365727669636541637469766174696F6E456E67696E653A312
E300000000000000002000000000000007000010200000000000D31302E323
2372E312E323031000022610000001B5374616E64617264496D706C4E616
D652F736165504F412F53414500000000020000000000000000800000000004
A414300000000010000001C0000000000010001000000010501000100010
1090000000105010001000000010000002C0000000000000001000000010
000001C00000000000100010000000105010001000101090000000105010
001
```

Monitoring NIC Proxies on Solaris Platforms

You can use MBeans to monitor NIC proxies. MBeans are a feature of the Java Management Extension (JMX) software. If you want to monitor the MBeans for NIC proxies, your Web application server must include a JMX agent.

NIC proxies create one instance of an MBean called NicProxyMgmt to provide information about the role of the NIC proxy to the JMX agent. The way you view the MBeans depends on the particular Web application server and the interfaces that its JMX agent provides. Table 15 shows the information that this MBean provides.

You can reset the values of many NicProxyMgmt MBean properties to zero.

To reset the NicProxyMgmt MBean properties to zero:

- Execute the reset counters operation through the mechanism that the JMX agent for your Web application server provides.

  Table 15 shows which counters the reset operation affects.

Table 15: Information That the NicProxyMgmt MBean Provides

Property	Description	Ability to Reset to Zero
nicProxyName	Name of the NIC proxy. Different NIC proxies may exist, providing different functionality.	No
numKeysCachedLocally	Number of key-value pairs that are cached in the NIC proxy (the bigger the cache, the less likely the NIC proxy will have to involve the distributed NIC components in lookups across the network).	No
numLookups	Number of times that the Web application containing this NIC proxy has requested the NIC proxy to look up a data key.	Yes
numLookupErrors	Number of lookups that have failed.	Yes

Table 15: Information That the NicProxyMgmt MBean Provides (continued)

Property	Description	Ability to Reset to Zero
numKeysNoMatch	Number of lookups in which the provided key does not map to any value.	Yes
numKeysOneMatch	Number of lookups in which the provided key maps to exactly one value.	Yes
numKeysMultiMatch	Number of lookups in which the provided key maps to more than one value.	Yes
lookupTimeAvg	For the 100 most recent (successful and unsuccessful) lookups, the average time (in milliseconds) of the lookup.	Yes
lookupTimeMin	For the 100 most recent (successful and unsuccessful) lookups, the minimum time (in milliseconds) of the lookup.	Yes
lookupTimeMax	For the 100 most recent (successful and unsuccessful) lookups, the maximum time (in milliseconds) of the lookup.	Yes

Chapter 16

Developing Applications That Use NIC

This chapter describes how to develop an external application to interact with a network information collector (NIC). Topics include:

- External Application Requirements for NIC on page 215
- External Non-Java Applications That Use NIC on page 215
- External Java Applications That Use NIC on page 216
- Updating Information About Address Pools on page 223

External Application Requirements for NIC

If you write an external application to use NIC to perform a resolution, you can include NIC functionality in one of the following ways:

- For non-Java applications, use the interface module NicAccess, an IDL file that provides access to the NIC locator feature. The NIC locator can resolve the value of one or more keys.
- For Java applications, include the NIC proxy client libraries to use NIC in client/server mode.
- For Java applications, include the NIC proxy client libraries and the NIC host client libraries to use NIC in local host mode.

External Non-Java Applications That Use NIC

If you write an application in a language other than Java, you can use the NIC access interface module, a simplified CORBA interface, to perform one or more resolutions. By using this interface you can access through CORBA NIC locators, NIC proxies that run within the NIC host. The configuration properties for NIC locators are similar to those for NIC proxies in applications such as aggregate services and the sample residential portal.

For information about the NIC access interface module, see the API documentation in the SRC software distribution in the folder *SDK/doc/idl/nic* or on the Juniper Networks Web site at

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

Creating a NIC Locator to Include with a Non-Java Application

A NIC locator provides the same functionality as a NIC proxy, but is designed to work with non-Java applications.

You use the NIC access interface module to include NIC locators with your application by compiling the IDL file with your application files.

To use the NIC access interface module to create NIC locators:

1. Connect to the directory.
2. Obtain a CORBA reference to the NIC access interface from one of the following:
   - The access IOR provided in the directory in the dynamic configuration DN under the hostname—typically, *host/demohost*.

     You can read this information from SDX Admin from a host under *ou = dynamicConfiguration, ou = Configuration, o = Management, o = umc.*
   - A corbaloc URL in the format:

     ```
     corbaloc::<host>:8810/Access
     ```
3. From the NIC access interface module, obtain a NIC locator, as identified by NicFeature. For example:

   ```
   feature = access.getLocatorFeature(nicNameSpace); //nicNameSpace example
   "/nicLocators/ip"
   ```

   In the NIC configuration scenarios, the syntax for a NIC locator is `/nicLocators/<NIC key type>` where.
   - `nicLocators`— Specifies all of the NIC locators in a NIC host.
   - `<NIC key type>`— Specifies the type of data that the key provides for the NIC resolution, such as ip, login, DN.
4. Search for the key. For example:

   ```
   feature.lookupSingle(NicLocatorKey key) //NicLocatorKey is coming from the IDL
   ```

For information about the NIC access interface module, see the API documentation in the SRC software distribution in the folder *SDK/doc/idl/nic* or on the Juniper Networks Web site at

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

External Java Applications That Use NIC

If you write an external Java application that interacts with a NIC, include NIC libraries in the application. These libraries are for NIC proxies and local NIC hosts. These libraries are located in the SRC distribution under *SDK/lib/nic*.

Typically, each NIC resolution process requires one NIC proxy. For example, the OnePopLogin sample data includes two resolution processes:

- Mapping of a subscriber's IP address to the subscriber's login name
- Mapping of the subscriber's login name to the SAE reference

An application that uses both these resolution processes would require two NIC proxies.

The NIC proxy provides a simple Java interface, the NIC application programming interface (API). You configure the NIC proxy to communicate with one resolver. For efficiency if you use NIC in client/server mode, the NIC proxy caches the results of resolution requests so it can respond to future requests for the same key without contacting the resolver.

The SRC software includes a factory interface, the NIC factory, to allow applications to instantiate, access, and remove NIC proxies. It also includes JAR files for NIC client and NIC host libraries.

Developing a Java Application to Communicate with a NIC Proxy

You must configure an application to communicate with a NIC proxy.

If you are using Java Runtime Environment (JRE) 1.3 or higher, you must include in your application the Java archive (JAR) files, which are in the SRC software distribution in the folder */SDK/lib/* with your application:

Configuration tasks that use the API calls to communicate with the NIC proxy are:

1. Instantiating a Configuration Manager on page 218
2. Passing a Reference to the Configuration Manager to the NIC Factory on page 218
3. Instantiating the NIC Factory Class on page 218
4. (Optional) Initializing Logging on page 219
5. Instantiating the NIC Proxy on page 220
6. Managing a Resolution Request on page 220
7. Deleting Invalid Results from the NIC Proxy's Cache on page 222
8. *Removing the NIC Proxies* on page 222

For more information about the API calls, see the online documentation in the SRC software distribution in the folder */SDK/doc/nic* or on the Juniper Networks Web site at

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

Instantiating a Configuration Manager

The application must instantiate a configuration manager.

To enable the application to instantiate a configuration manager to obtain a NIC instance from the NIC factory:

- Call one of the following methods:
    - For some applications (other than Web applications), in which you must define the system property -DConfig.bootstrapFilename, you can call the following method:

      `ConfigMgr configMgr = ConfigMgrFactory.getConfigMgr();`

    - For Web applications, you can instantiate the configuration manager as follows:

      `ConfigMgr configMgr = ConfigMgrFactory.getConfigMgr(properties);`

        - properties—java.util.Properties object, typically the bootstrap file, which contains all the configuration properties for the NIC proxy.

Passing a Reference to the Configuration Manager to the NIC Factory

To pass a reference to the configuration manager to the NIC factory class:

- Call the following method in the application:

  `NicFactory.setConfigManager(configMgr);`

Instantiating the NIC Factory Class

The way you instantiate the NIC factory depends on the object request broker (ORB) configuration:

- If the NIC proxy uses the default ORB, call the following method in the application:

  `NicFactory nicFactory = NicFactory.getInstance();`

  This code instantiates a new NIC factory. Unless the NicFactory.destroy method has been called, subsequent calls to this method will return the instantiated NIC factory.

- If the NIC proxy does not use the default ORB, call the following method:

  ```
  NicFactory.initialize(props);
  NicFactory nicFactory = NicFactory.getInstance();
  ```

  - props—java.util.Properties object, which contains the ORB properties for the NIC proxy. For example, if the NIC proxy uses JacORB but JacORB is not the default ORB, the ORB properties are:

    ```
    org.omg.CORBA.ORBClass=org.jacorb.orb.ORB
    org.omg.CORBA.ORBSingletonClass=org.jacorb.orb.ORBSingleton
    ```

  This code will instantiate a new NIC factory using the specified ORB. Unless the application has called the NicFactory.destroy method, subsequent calls to the getInstance() method will return the instantiated NIC factory. However, if the application has called the destroy() method, it must recall the initialize() method before it can call the getInstance() method.

  For information about the NicFactory.destroy method, see *Removing the NIC Proxies* on page 222.

Initializing Logging

You must initialize logging only if you want to view the logging information produced by the NIC proxy.

To enable the application to initialize logging:

- Call the following method:

  ```
  Log.init(configMgr, configNameSpace);
  ```

  - configMgr—Instance of the configuration manager, the value returned from the getConfigMgr() method
  - configNameSpace—String that specifies the configuration namespace where you defined the logging properties
    - ☐ If you define the logging properties in the bootstrap file, specify the root namespace, "/".

      ```
      Log.init(configMgr, "/");
      ```

    - ☐ If you define the logging properties in the directory, specify the namespace relative to the property Config.net.juniper.smgt.lib.config.staticConfigDN, which you configure in the bootstrap file.

      ```
      Log.init(configMgr, "/Applications/Quota");
      ```

Instantiating the NIC Proxy

To enable the application to instantiate a NIC proxy:

- Call the following method:

  ```
  NIC nicProxy = nicFactory.getNicComponent(nicNameSpace, configMgr)
  ```

  Alternatively, if the expected data value (specified for the property nic.value in the NIC proxy configuration) is an SAE reference, you can call the following method:

  ```
  SaeLocator nicProxy = nicFactory.getSaeLocator(nicNameSpace, configMgr);
  ```

  - nicFactory—Instance of the NIC factory
  - nicNameSpace—String that specifies the configuration namespace where you defined the properties for the NIC proxy
    - If you define the NIC properties in the bootstrap file, specify the root namespace, "/".

      ```
      NIC nicProxy = nicFactory.getNicComponent("/", configMgr)
      ```

    - If you define the properties in the directory, specify the namespace relative to the property Config.net.juniper.smgt.lib.config.staticConfigDN, which you specified in the bootstrap file.

      ```
      NIC nicProxy = nicFactory.getNicComponent("/Applications/Quota",
      configMgr)
      ```

  - configMgr—Instance of the configuration manager, the value returned from the getConfigMgr() method

Managing a Resolution Request

To enable the application to submit a resolution request and obtain the associated values:

1. Construct a NicKey object to enable the application to pass the data key to the NIC proxy:

   ```
   NicKey nicKey = new NicKey(stringKey);
   ```

   - stringKey—Data key for which you want to find corresponding values.

     For the syntax of allowed data types, see *Chapter 17, NIC Resolution Process*.

2. If the resolution process specifies constraints that you wish to provide in the resolution request, add them to the NicKey object:

   ```
   NicKey.addConstraint(constName, constValue);
   ```

   - constName—Name of the constraint.

     For the allowed data types and their syntax, see *Chapter 17, NIC Resolution Process*.

   - constValue—Specific value of the constraint.

     For the allowed syntax for the data types, see *Chapter 17, NIC Resolution Process*.

3. Call a method that starts the resolution process.

   For example, you can call a method specified in the NIC interface:

   ```
   NicValue val = nicProxy.lookupSingle(nicKey);
   ```

   Alternatively, if the expected data value is an SAE reference, you can call the following method:

   ```
   SaeId saeId = nicProxy.lookupSae(nicKey);
   ```

4. Call the getValue method to access the string representation of the data value obtained by the NIC proxy.

   ```
   String val=val.getValue();
   ```

   Alternatively, if the expected data value is an SAE reference:

   ```
   String val=saeId.getValue();
   ```

5. (Optional) Call a method to get intermediate values obtained during a resolution.

   - Call the getIntermediateValue method if the application expects only one value. This method takes the name of a data type and returns as a string the first value it finds.

     ```
     String getIntermediateValue(String dataTypeName){};
     ```

     For information about data types, see *Chapter 17, NIC Resolution Process*.

- Call the getIntermediateValues or getAllIntermediateValues method if the application expects multiple values. These methods take the name of a data type and return values as follows:
  - The getIntermediateValues method returns a list of values as a string array.

    `String[] getIntermediateValues(String dataTypeName){};`

  For information about data types, see *Chapter 17, NIC Resolution Process*.

  - The getAllIntermediateValues method returns a map of all intermediate values for the request. The key for the map is the name of the network data type, and the value of the map is a string array of the intermediate values.

    `Map getAllIntermediateValues();`

Deleting Invalid Results from the NIC Proxy's Cache

If the application receives an exception when using values that the NIC proxy returned for a specific key, it must inform the NIC proxy to delete this entry from its cache.

To enable the application to inform the NIC proxy to delete an entry from its cache:

- Call the following method:

  `nicProxy.invalidateLookup(nicKey, nicValue);`

  - nicKey—Data key that you want to remove from the cache
  - nicValue—Optional data value that corresponds to this key

    If the application passes a null data value to the NIC proxy, the NIC proxy removes all the values associated with the data key from its cache.

Removing the NIC Proxies

Make sure that before your application shuts down, it removes the NIC proxy instances to release resources for other software processes.

To remove one NIC proxy instance:

- Call the following method:

  `NicProxy.destroy();`

  To remove all NIC proxy instances, call the following method:

  `NicFactory.destroy();`

Updating Information About Address Pools

If you associate an existing address pool with an interface and you do not want to wait for this new information to be propagated based on the Cache Entry Age property of the NIC proxy or the Event Life Expectancy property of the agents, then you must manually clear the NIC proxy cache.

To clear the NIC proxy cache when an application is deployed in a J2EE container that supports Java Management Extension (JMX) software, do one of the following:

- Use the NicProxyMgmt MBean.
- Restart the application.
- Restart the application server.

For information about modifying the NIC proxy cache properties, see *Chapter 13, Configuring Applications to Communicate with an SAE*.

Chapter 17

NIC Resolution Process

This chapter provides information about the NIC resolution process. You should be familiar with this information if you customize a NIC configuration scenario. Topics include:

- Overview of the NIC Resolution Process on page 225
- NIC Data Types on page 227
- Constraints as NIC Data Types on page 229

Overview of the NIC Resolution Process

Because NIC can process all types of network data, you must use different resolution processes for different types of data mappings to maximize the performance of the NIC configuration. Resolving data requests consumes significant resources.

Table 16 shows the resolutions that the components in the NIC configuration scenarios perform. For customized types of resolutions, contact Juniper Networks Professional Services.

Table 16: Available NIC Resolutions

Key	Value
Subscriber’s IP address (JUNOS routing platform)	SAE reference
Subscriber’s IP address	Subscriber’s login name
Subscriber’s IP address	SAE reference
Subscriber’s login name	SAE reference
Subscriber’s username	SAE reference
Access DN	SAE reference

NIC Realms

Each resolution process and the resolvers that perform that process are defined by a *realm*—a group of resolvers that perform a series of resolution tasks to provide a mapping from a specified key to a specified data type. For example, the sample data provided for the NIC includes a realm called dn in which the resolution process takes an access subscriber's distinguished name (DN) as the key and returns a reference to the SAE managing this subscriber as the value.

A set of hosts in a NIC can support multiple realms. Similarly, the agents in a NIC can support more than one realm. However, you can assign a resolver only to one realm.

A NIC host can support NIC resolvers for multiple realms. Consequently, you can simplify the NIC configuration and minimize the use of network resources by limiting the number of NIC hosts in your NIC configuration. NIC hosts can also handle multiple NIC resolvers in the same realm. In this case, when a NIC host receives a request, it chooses a NIC resolver as follows:

1. It identifies the NIC resolvers that are available to process the request.

2. If multiple NIC resolvers are available, it obtains a cost value associated with the resolution process from each resolver and selects the resolver that has the lowest cost value.

Key to Value Resolution

A resolution process typically defines several transitions or *roles*, with each transition resolving a NIC key to a value. For example, the resolution process to identify the SAE that manages a particular subscriber based on that subscriber's IP address involves the following roles:

1. Given the IP address, determine the IP address pool.

2. From the IP address pool, determine the VR.

3. From the VR, determine the SAE that manages that VR.

A role specifies the types of data with which it works. NIC supports a number of data types, including one that lets you add an identifier to other data types to let you specify different values for one data type.

For information about NIC data types, see *NIC Data Types* on page 227 and *Constraints as NIC Data Types* on page 229.

NIC Data Types

The NIC supports the data types that appear in the following list. You can qualify these data types by adding an identifier to:

- Distinguish between different instances of a data type in a resolution scenario.
- Provide information about a data type to clarify the use of that data type in a resolution.

For information about qualifying data types, see *Chapter 18, Customizing a NIC Configuration.*

AnyString

- Generic data type to represent the information that you want to collect.
- Value—Alphanumeric characters
- Guidelines—You can qualify this data type with an identifier to provide information about the type of data that AnyString represents.
- Example—My(IP), My(Vr)

Dn

- DN of an access.
- Value—DN
- Example—*accessName = PrimaryAccess, enterpriseName = juniper, ou = Sunnyvale, retailerName = VPNprovider, o = Users, o = umc*

Domain

- Domain name.
- Value—Name of a domain
- Example—Example.net

Enterprise

- DN of an enterprise.
- Value—DN
- Example—*enterpriseName = juniper, ou = Sunnyvale, retailerName = VPNprovider, o = Users, o = umc*

Router

- Name of router.
- Value—Text string
- Example—router1

Interface

- Name of a router's interface.
- Value—<interfaceName>@<vrName>@ <routerName>
  - <interfaceName>—Name of the interface, such as fastEthernet 3/1, exactly as it is configured on the router
  - <vrName>—Name of VR exactly as it is configured on the router
  - <routerName>—Name of router exactly as it is configured on the router
- Example—FastEthernet4/1.0@boston@router1

InterfaceId

- Identifier of an interface.
- Value—<intfIndex>@<routerName>
  - <intfIndex>—Simple Network Management Protocol (SNMP) index of the interface
  - <routerName>—Name of router exactly as it is configured on the router
- Example—4@router1

Ip

- Subscriber's IP address.
- Value—IP address
- Example—192.0.2.10

IpPool

- IP address pool.
- Value—Range of IP addresses enclosed in square brackets and parentheses
- Guidelines—If you enter an IP address that includes a value greater than 255 in one octet of the address, that part of the address is masked to fit the eight bits.
- Example—([192.0.2.0 192.0.2.255])

SaeId

- SAE reference.
- Value—CORBA interoperable object reference (IOR) for SAE
- Example—IOR:000000000000002438444C3A736...

Vr

- Name of the virtual router.
- Value—<vrName>@<routerName>
  - <vrName>—Name of VR exactly as it is configured on the router
  - <routerName>—Name of router exactly as it is configured on the router
- Example—vr1@router1

Constraints as NIC Data Types

Constraints are data types that a resolver uses when it executes a role. You can define:

- Multiple constraints for a role—Software performs an OR operation to determine whether the constraint is met.
- Multiple data types in a constraint—Software performs an AND operation to determine whether the multiple constraints are met.

Constraints can be either mandatory or optional. If a constraint is mandatory and the resolver for the role does not receive an appropriate value in the data request, the resolver must obtain the constraint value from other NIC resolvers. However, if a constraint is optional and the resolver for the role does not receive an appropriate value in the data request, the resolver can execute its role without the constraint value. In this case, the resolver may obtain multiple values for the data key, and the NIC host responds to the NIC proxy as follows:

- If the request is for multiple results, the host provides all the results.
- If the request is for one result and the resolution process returns different results, the host returns an error message.
- If the resolution process returns multiple instances of the same result, the resolver provides only one result.

For example, if you want to obtain an SAE reference for a subscriber's IP address, you could define the following roles:

1. From the IP address, determine the VR (mandatory constraint IpPool).
2. From the VR, determine the SAE that manages that VR.

Because the first step has a mandatory constraint, the resolver for this role must use the IP pool supplied in the request, or obtain the IP pool from another resolver that determines IP pools from IP addresses. So you must define an extra step at the start of the resolution process:

1. From the IP address, determine the IP pool.
2. From the IP address, determine the VR (mandatory constraint IpPool).
3. From the VR, determine the SAE that manages that VR.

Chapter 18

Customizing a NIC Configuration

In most cases, you use the network information collector (NIC) configuration scenarios supplied with the SRC software. If you need a different resolution scenario, you can customize these scenarios for a NIC that runs on a Solaris platform. Topics include:

Before You Customize a NIC Configuration

We recommend that you customize a NIC configuration by adding components to or removing components from a NIC configuration scenario that is supplied with the sample data.

Before you customize a NIC configuration:

- Review the configuration scenarios provided with the SRC software to determine whether you must use a NIC configuration scenario different from the ones supplied.

  See *Chapter 9, Locating Subscriber Information with the NIC*.

- Make sure that you are familiar with the NIC resolution process.

  See *Chapter 17, NIC Resolution Process*.

Planning a Custom NIC Configuration

Typically, you combine components from existing NIC configuration scenarios to create a new one.

To plan how to combine NIC configuration scenarios:

1. Review the NIC configuration scenarios available in the sample data, and decide which scenarios provide resolutions that you want to use. See *Chapter 9, Locating Subscriber Information with the NIC*.
2. In SDX Admin, review the configuration for each scenario to be used. The configuration scenarios appear under *l = NIC, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*.

   If both scenarios use agents that have the same name, review the agent configuration to see whether or not the configurations are the same.
3. Decide whether a non-Java application is to communicate with NIC host.
4. Make a list of which agents and resolvers (stored under *l = realms*) to use from each configuration scenario.
5. Decide which configuration scenario to use as a base. You will make a copy of this scenario and modify it.

Creating a Custom NIC Configuration by Adding Components to an Existing Scenario

You can create a custom NIC configuration scenario in SDX Admin by copying an existing configuration scenario and modifying it.

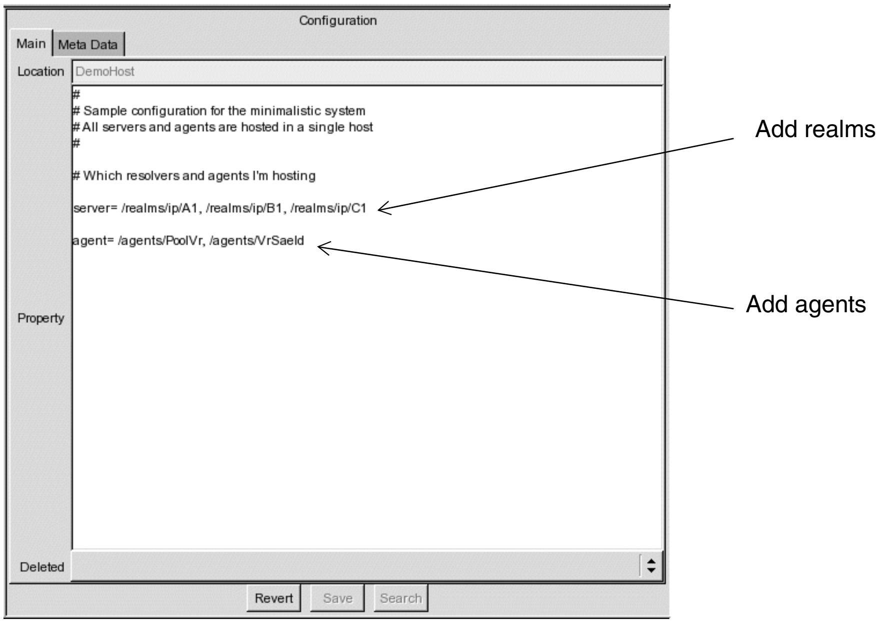
To use SDX Admin to customize a NIC configuration:

1. In the navigation pane, select a NIC configuration scenario under *l = NIC, ou = staticConfiguration, ou = Configuration, o = Management, o = umc* to supply the basic type of resolution.
2. Expand the configuration scenario; then expand the agents item and the realms item for the configuration scenario, and review the configuration for each agent and each realm.
3. In the navigation pane, select a NIC configuration scenario under *l = NIC, ou = staticConfiguration, ou = Configuration, o = Management, o = umc* that provides other types of resolutions that you want to add to the first configuration scenario.
4. Expand the configuration scenario; then expand the agents item and the realms item for the configuration scenario, and review the configuration for each agent and each realm.

5. Make a copy of the base resolution scenario:
    a. Right-click the configuration scenario under *l = NIC, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*, and select **Copy Tree**.
    b. Right-click the *l = NIC* item, and select **Paste** (Configuration).
    c. In the Paste Configuration dialog box, enter the name of the custom scenario in the Location box, and click **OK**.
    d. Expand the configuration for the custom scenario and for the agents, hosts, and realms under that scenario.
6. Copy agents to be added to this configuration scenario from another configuration scenario. For each agent to be added:
    a. Right-click the agent in the configuration scenario that is to be copied, and select **Copy Tree**.
    b. Right-click the agent item under the configuration scenario, and select **Paste** (Configuration).
    c. If the name of the agent does not appear in the list of agents, in the Paste Configuration dialog box keep the name of the Location the same, and click **OK**.

       If the name of the agent already appears in the list of agents and the configuration for this agent is different, in the Paste Configuration dialog box, enter a new name for the agent in the Location field, and click OK.
7. Copy each resolution to be added to the configuration scenario:
    a. Right-click the resolution under *l = realms* in configuration scenario that is to be copied, and select **Copy Tree**.
    b. Right-click *l = realms* under the new configuration scenario, and select **Paste** (Configuration).
    c. In the Paste Configuration dialog box, keep the name of the location the same, and click **OK**.
8. If a non-Java application is to use the configuration scenario, copy the NIC locator configuration for each resolution:
    a. Right-click the NIC locator under *l = nicLocators* in the configuration scenario that is to be copied, and select **Copy Tree**.
    b. Right-click *l = nicLocators* under the new configuration scenario, and select **Paste** (Configuration).
    c. In the Paste Configuration dialog box, keep the name of the location the same, and click **OK**.

9. Expand **hosts**, and select **DemoHost**. In the content pane, add entries for the agents added and the realms added.



Location: DemoHost

```
#
# Sample configuration for the minimalistic system
# All servers and agents are hosted in a single host
#

# Which resolvers and agents I'm hosting

server= /realms/ip/A1, /realms/ip/B1, /realms/ip/C1

agent= /agents/PoolVr, /agents/VrSaeId
```

10. Click **Save**.

11. If you make changes to a directory entry that result in the entry being removed from its search filter, stop and then restart the NIC host.

    See *Managing Directory Changes for the Directory Agent* on page 236.

Creating a Custom NIC Configuration by Removing Components in an Existing Scenario

You can create a custom NIC configuration scenario in SDX Admin by copying an existing configuration and removing components from it. For example, you can remove IP resolution from the OnePopDnSharedIP scenario to create a CustomOnePopDn scenario.

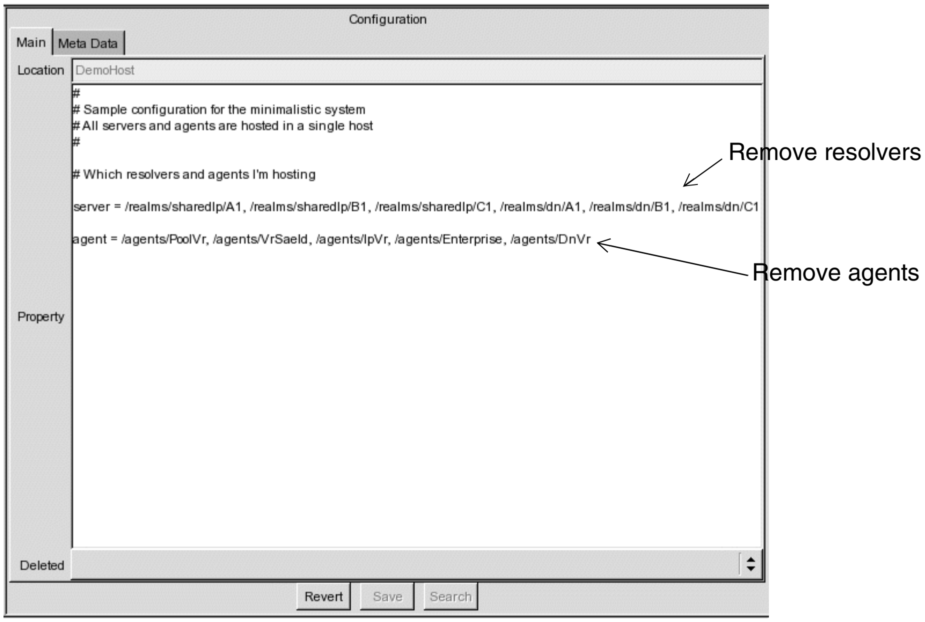
To use SDX Admin to customize a NIC configuration by removing components:

1. In the navigation pane, select the NIC configuration scenario under *l = NIC, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*.

2. Expand the configuration scenario; then expand the agents item and the realms item for the configuration scenario, and review the configuration for each agent and each realm.

3. To delete agents and resolvers that you want to remove, right-click the item and select **Delete**.

4. Expand **hosts**, and select **DemoHost**. In the content pane, remove entries for the agents and the resolvers that were deleted.



Location DemoHost

```
#
# Sample configuration for the minimalistic system
# All servers and agents are hosted in a single host
#

# Which resolvers and agents I'm hosting

server = /realms/sharedIp/A1, /realms/sharedIp/B1, /realms/sharedIp/C1, /realms/dn/A1, /realms/dn/B1, /realms/dn/C1

agent = /agents/PoolVr, /agents/VrSaeId, /agents/IpVr, /agents/Enterprise, /agents/DnVr
```

Remove resolvers

Remove agents

5. Click **Save**.

6. If you make changes to a directory entry that result in the entry being removed from its search filter, stop and then restart the NIC host.

   See *Managing Directory Changes for the Directory Agent* on page 236.

Qualifying NIC Data Types

If a NIC configuration scenario uses the same data type in more than one place in a resolution, you can add an identifier to the data type to clarify the source of the data type.

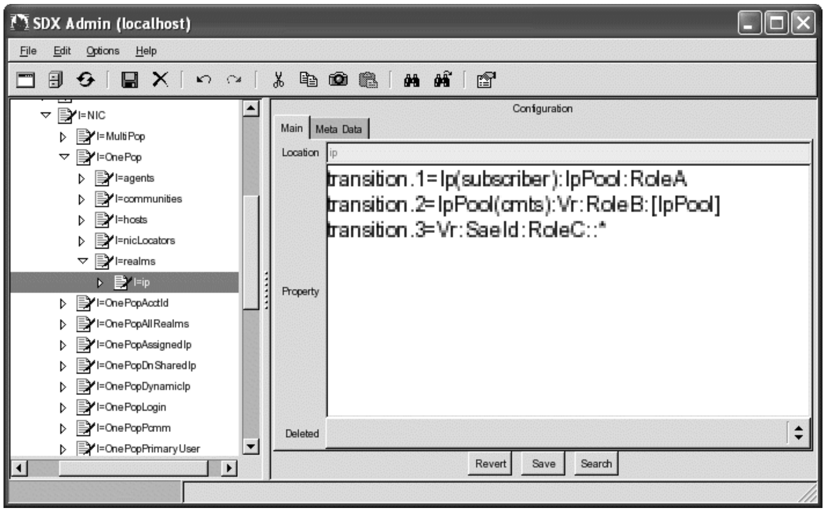
To use SDX Admin to qualify a data type:

1. In the navigation pane, select a resolution for a realm for a NIC configuration under *l = NIC, ou = staticConfiguration, o = Management, o = umc*.

   For example; *l = ip, l = realms, l = OnePop, l = NIC, ou = staticConfiguration, o = Management, o = umc.*

2. Directly after a data type, add an identifier for the value by adding the value after the data type in parentheses.

   In the following example, subscriber qualifies the IP data type on the transition.1 line, and cmts qualifies the IpPool data type on the transition.2 line.



Location: ip

Property:

```
transition.1=Ip(subscriber):IpPool:RoleA
transition.2=IpPool(cmts):Vr:RoleB:[IpPool]
transition.3=Vr:SaeId:RoleC::*
```

3. Click **Save**.

Managing Directory Changes for the Directory Agent

The NIC directory agent does not support dynamic changes to a directory entry that result in the entry’s being removed from its search filter.

To have a NIC directory agent recognize a change in which a directory entry is removed from its search filter:

- Restart the NIC host that contains the agent.

For example, consider the MultiPop scenario provided as part of the NIC sample data. If you remove the POP-Ottawa scope from the directory entry with the following DN:

```
virtualRouterName=default, orderedCimKeys=Ottawa_ERX_Node, o=Network, o=umc
```

then the OttawaPoolVr and OttawaVrSaeId agents will not dynamically detect the change. You must restart OttawaHost for the changes to take effect.

Chapter 19

NIC Configuration Scenarios

This chapter provides detailed descriptions of the network information collector (NIC) configuration scenarios. Topics include:

Overview of NIC Configuration Scenarios

The NIC configuration scenarios in the sample data provide resolutions for a variety of network configurations.

Each NIC scenario includes two types of configuration:

- Centralized—A single host configuration for use with NIC replication. In a centralized configuration all agents and resolvers reside on one host. The name of this host is DemoHost.
- Distributed—A multiple host configuration in which agents and resolvers are distributed among more than one host. This type of configuration is designed for use with NIC host redundancy. In most cases, the hosts are named OnePopH1 (a host in a pop) and OnePopBO (a host in a back office).

The best way to view the sample data is with the NIC Web Admin tool.

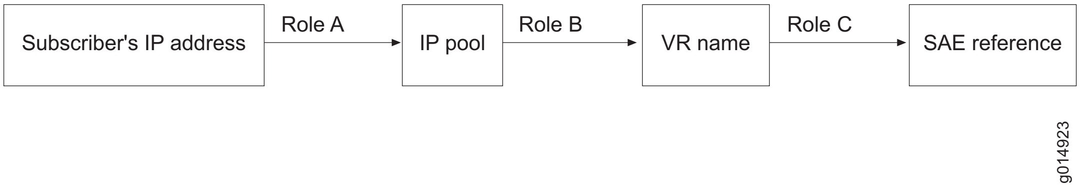
For a summary of the NIC configuration scenarios included in the sample data, see *Chapter 9, Locating Subscriber Information with the NIC*.

OnePop Scenario

The OnePop scenario illustrates a configuration that supports one POP. The realm for this configuration accommodates the situation in which IP address pools are configured locally on each VR. 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.

Figure 12 shows the resolution graph for this realm.

Figure 12: Resolution Process for ip Realm



The following agents collect information for resolvers in this realm:

- Directory agent PoolVr collects and publishes information about the mappings of IP address pools to VRs.
- Directory agent VrSaeId collects and publishes information about the mappings of VRs to SAEs.

The OnePop sample provides two host configurations: a centralized configuration and a distributed configuration. The OnePop Centralized configuration also provides an example of NIC host redundancy.

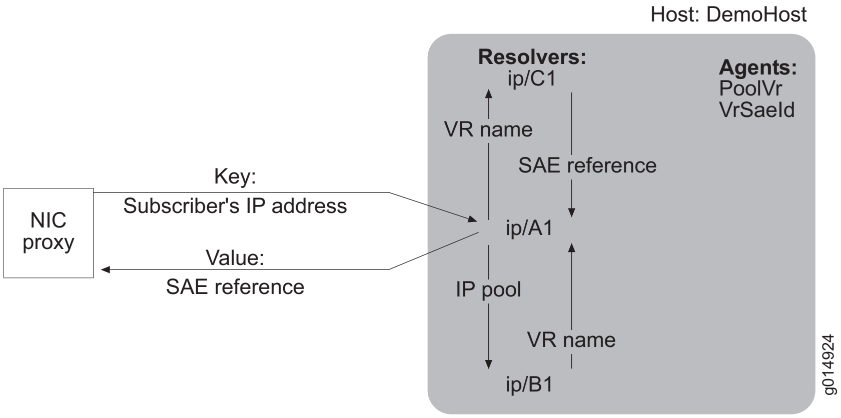
Centralized Configuration

In this configuration, single host DemoHost supports all agents and resolvers. When the NIC proxy sends a subscriber's IP address to host DemoHost, the following sequence of actions occurs:

1. The host passes the IP address to resolver A1.
2. Resolver A1 obtains an IP pool for the IP address and forwards the request to resolver B1.
3. Resolver B1 obtains a VR name for the IP pool and returns the VR name to resolver A1.
4. Resolver A1 forwards the VR name to resolver C1.
5. Resolver C1 obtains an SAE reference for the VR and returns the VR identity to resolver A1.
6. Resolver A1 passes the SAE reference to its host.
7. The host returns the SAE reference to the NIC proxy.

Figure 13 shows the interactions of the NIC components for this realm.

Figure 13: OnePop Centralized Configuration

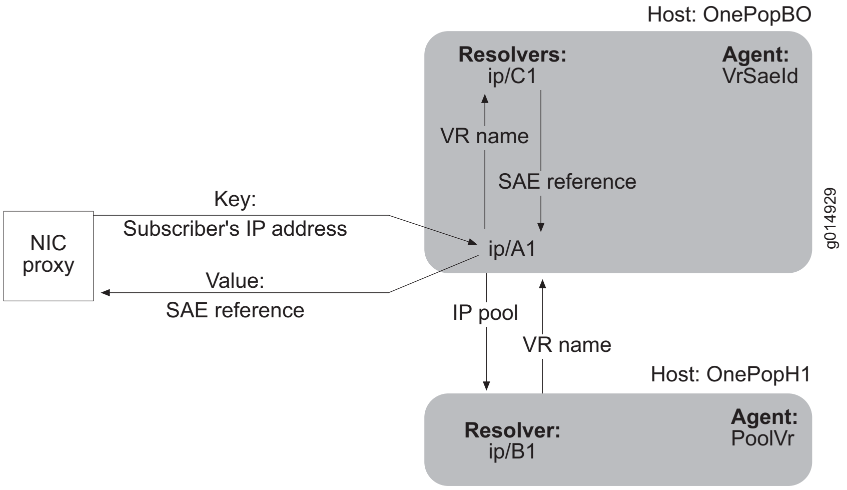


Distributed Configuration

In this configuration, the agents and resolvers are distributed among several hosts. When the NIC proxy sends a subscriber's IP address to host OnePopBO, the components execute the same actions as they do in the centralized configuration (see *Centralized Configuration* on page 239).

Figure 14 illustrates the interactions of the NIC components for this realm.

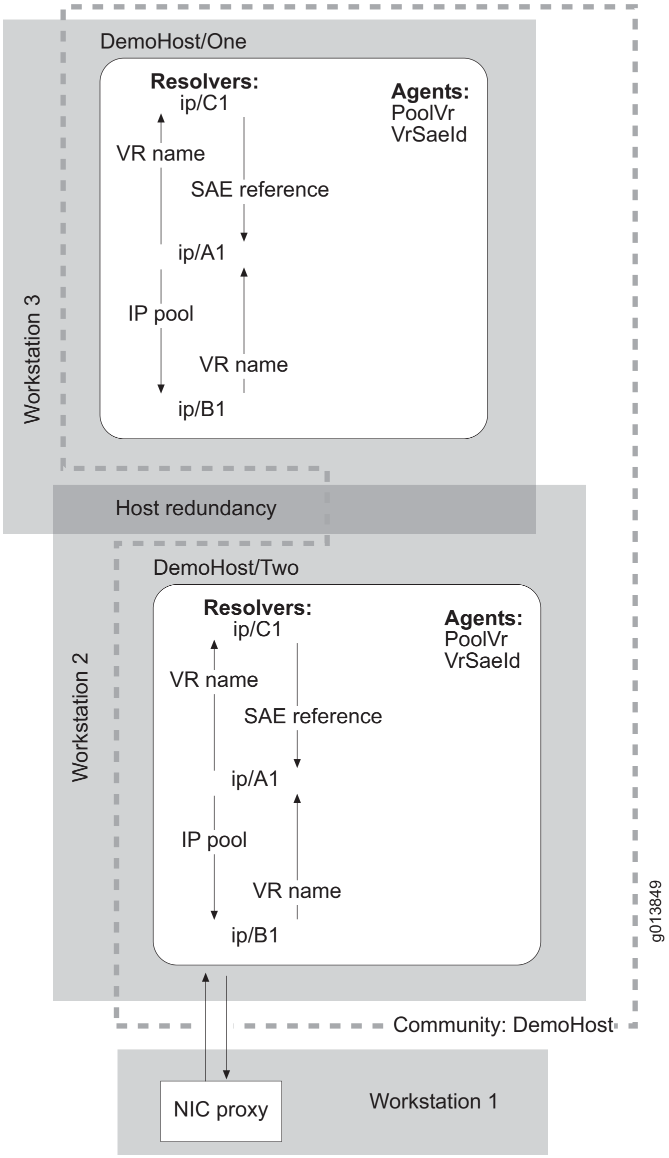
Figure 14: OnePop Distributed Configuration



Redundancy

This sample data includes host redundancy for the centralized configuration. The hosts DemoHost/One and DemoHost/Two, which are installed on different machines, provide host redundancy. These hosts form the community DemoHost, which does not include a monitor.

Figure 15: Redundancy for OnePop Centralized Configuration



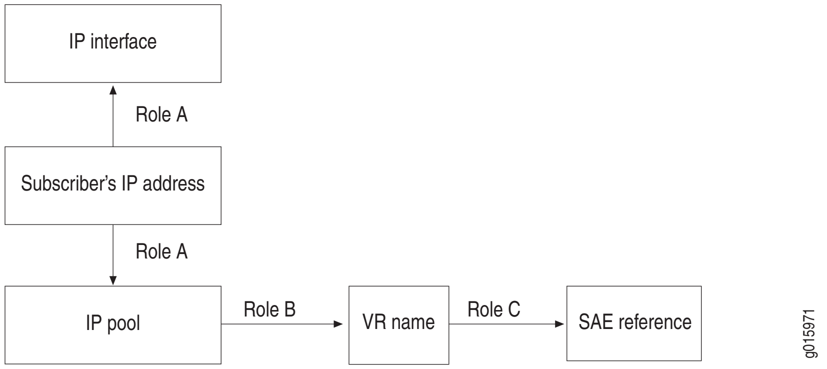
OnePopPcmm Scenario

This scenario is similar to the OnePop configuration scenario. It illustrates a configuration in which an assigned subscriber IP address managed by a network device such as a cable modem termination system (CMTS) device resolves to a reference to the SAE managing this subscriber. In this situation, the SAE acts as an application manager and interacts with the CMTS through a policy server.

The OnePopPcmm configuration scenario supports a PacketCable Multimedia Specification (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 realm for this configuration accommodates the situation in which IP pools are configured locally on each application manager group object. These IP pools represent an IP pools-managed policy decision point (PDP) group for one or more CMTS devices.

Figure 16 shows the resolution graph for this realm.

Figure 16: Resolution Process for Pcmm_am Realm



This scenario uses the same agents as the OnePop scenario. For the OnePopPcmm configuration scenario, the agent collects information from the application manager object instead of the virtual router entry. A virtual router name is generated in the format "default"@<pdpGroup>.

The OnePopPcmm scenario provides two host configurations: a centralized configuration and a distributed configuration.

Centralized Configuration

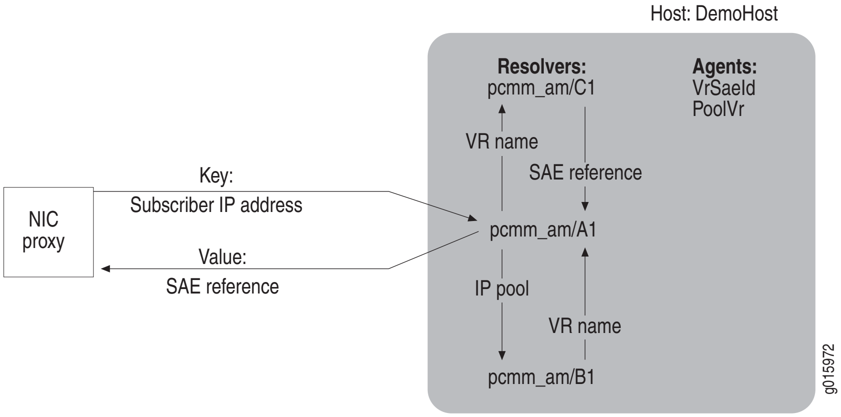
In this configuration, the single host DemoHost supports all agents and resolvers. When a NIC proxy sends a subscriber's IP address to host DemoHost, the following sequence of actions occurs:

1. The host passes an assigned subscriber IP address resolver A1.
2. Resolver A1 obtains the IP pool name and the interface name, and forwards the request to resolver B1.
3. Resolver B1 obtains the VR name for the IP pool name and interface name, and returns the VR name to resolver A1.

4. Resolver A1 forwards the VR name to resolver C1.
5. Resolver C1 obtains an SAE reference for the VR and returns it to resolver A1.
6. Resolver A1 passes the SAE reference to its host.
7. The host returns the SAE reference to the NIC proxy.

Figure 17 show the interactions of the NIC components for this realm.

Figure 17: OnePopPcmm Centralized Configuration

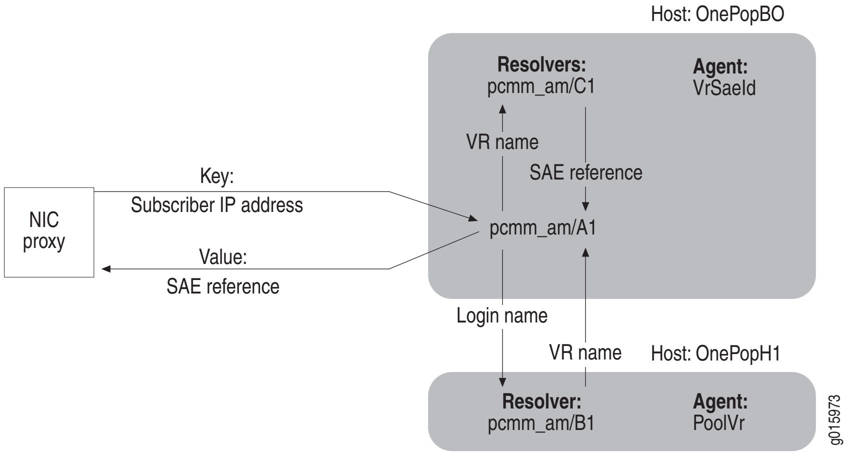


Distributed Configuration

In this configuration, the agents and resolvers are distributed among two hosts. When the NIC proxy sends a subscriber's IP address to host OnePopBO, the components execute the same actions as they do in the centralized configuration (see *Centralized Configuration* on page 242).

Figure 18 illustrates the interactions of the NIC components for this realm.

Figure 18: OnePopPcmm Distributed Configuration



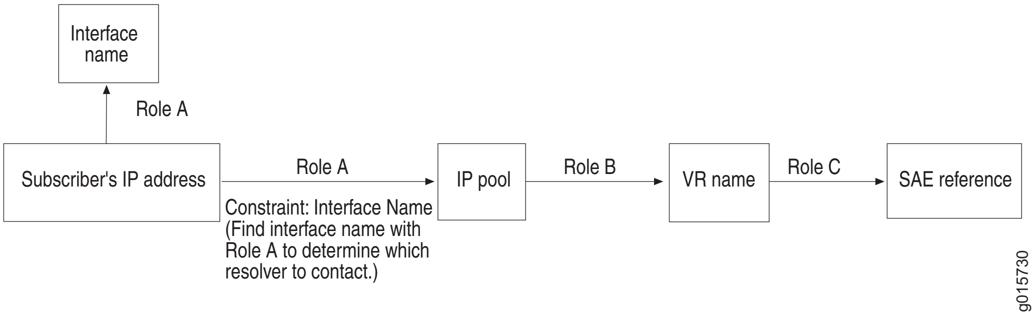
OnePopDynamicIp Scenario

This scenario illustrates a configuration that is very similar to the OnePop scenario. The realm for this configuration accommodates the situation in which IP address pools are configured locally on each virtual router 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 scenario supports a configuration scenario for a PacketCable Multimedia Specification (PCMM) environment in which you use the assigned IP subscriber method to log in subscribers, and use the NIC to determine the subscriber's SAE. In this scenario, the SAE acts as a combined application manager and policy server; it directly manages CMTS devices.

Figure 19 shows the resolution graph for this realm.

Figure 19: Resolution Process for dynamicIp Realm


Interface name
Role A
Subscriber's IP address
Role A
Constraint: Interface Name (Find interface name with Role A to determine which resolver to contact.)
IP pool
Role B
VR name
Role C
SAE reference
g015730

The following agents collect information for resolvers in this realm:

- Directory agent PoolVr collects and publishes information about the mappings of IP address pools to VRs.
- Directory agent VrSaeId collects and publishes information about the mappings of VRs to SAEs.

The OnePopDynamicIp scenario provides two host configurations: a centralized configuration and a distributed configuration.

Centralized Configuration

In this configuration, single host DemoHost supports all agents and resolvers. When the NIC proxy sends a subscriber's IP address to host DemoHost, the following sequence of actions occurs:

1. The host passes the IP address to resolver A1.
2. Resolver A1 obtains an IP pool name and interface name for the IP address, and forwards the request to resolver B1.
3. Resolver B1 obtains a VR name for the IP pool name and interface name, and returns the VR name to resolver A1.
4. Resolver A1 forwards the VR name to resolver C1.

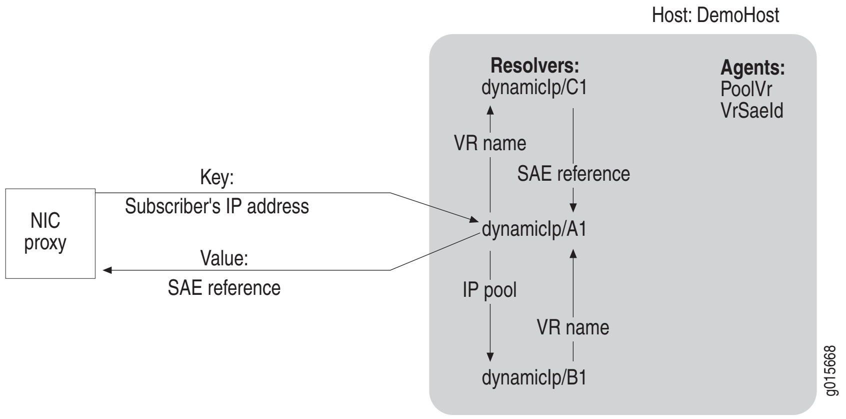
5. Resolver C1 obtains an SAE reference for the VR and returns the VR identity to resolver A1.

6. Resolver A1 passes the SAE reference to its host.

7. The host returns the SAE reference to the NIC proxy.

Figure 20 illustrates the interactions of the NIC components for this realm.

Figure 20: OnePopDynamicIp Centralized Configuration

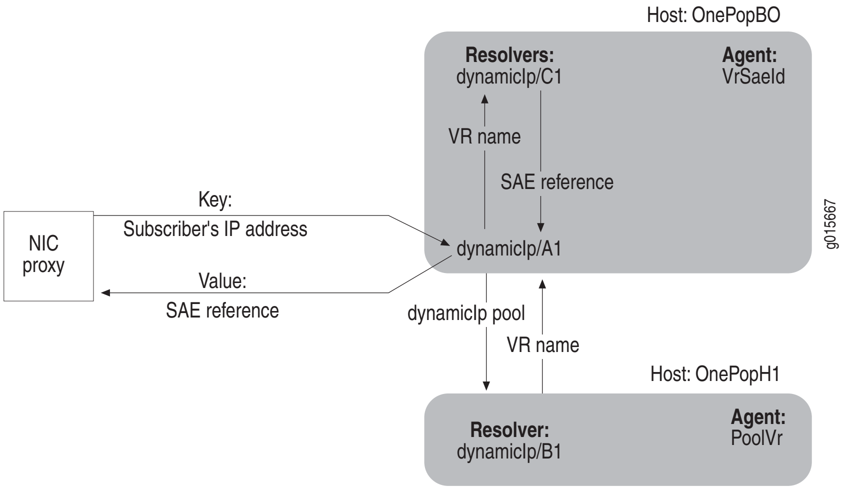


Distributed Configuration

In this configuration, the agents and resolvers are distributed among several hosts. When the NIC proxy sends a subscriber's IP address to host OnePopBO, the components execute the same actions as they do in the centralized configuration (see *Centralized Configuration* on page 244).

Figure 21 illustrates the interactions of the NIC components for this realm.

Figure 21: OnePopDynamicIp Distributed Configuration

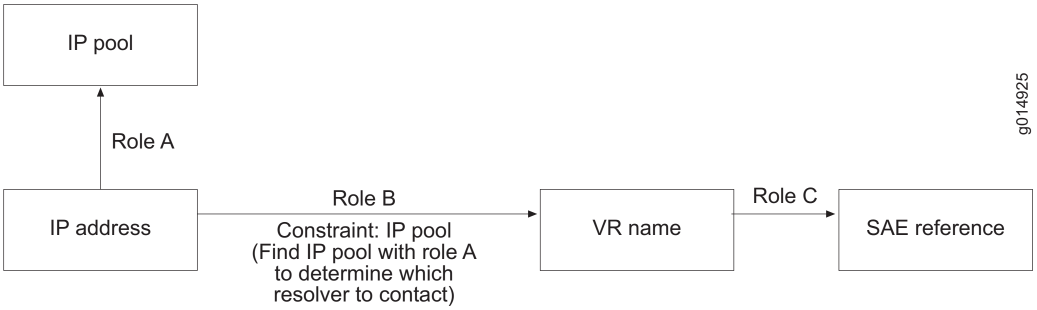


OnePopSharedIp Scenario

This scenario illustrates a configuration that is very similar to the OnePop scenario. However, the realm for this configuration accommodates the situation in which IP address pools are shared by VRs in the same POP. 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.

Figure 22 shows the resolution graph for this realm.

Figure 22: Resolution Process for sharedIp Realm


IP pool
Role A
IP address
Role B
Constraint: IP pool (Find IP pool with role A to determine which resolver to contact)
VR name
Role C
SAE reference
g014925

The following agents interact with resolvers in this realm:

- SAE plug-in agent IpVr collects and publishes information about the mappings of IP addresses to VRs.
- Directory agent PoolVr collects and publishes information about the IP address pools used by the VRs in a POP. Because the IP address pools are shared between VRs, this agent discards information about VRs.
- Directory agent VrSaeId collects and publishes information about the mappings of VRs to SAEs.

The OnePopSharedIP scenario provides two host configurations: a centralized configuration and a distributed configuration.

Centralized Configuration

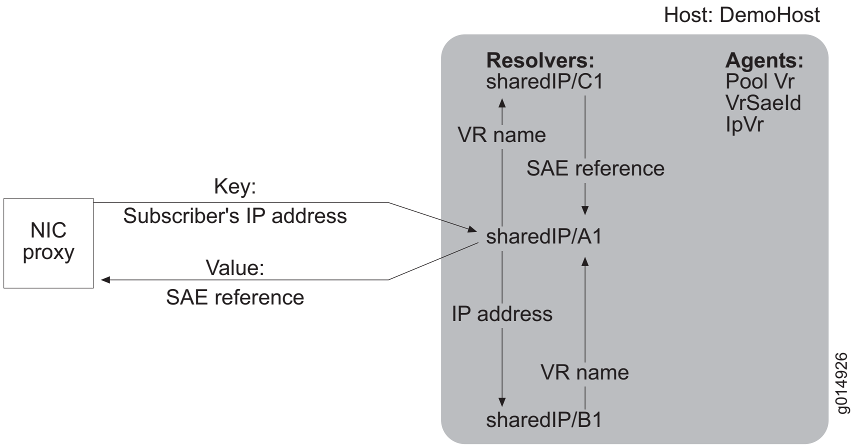
In this configuration, single host DemoHost supports all agents and resolvers. When the NIC proxy sends a subscriber's IP address to host DemoHost, the following sequence of events occurs:

1. The host passes the IP address to resolver A1.
2. Resolver A1 obtains an IP pool for the IP address.
3. Resolver A1 forwards the IP address and the IP pool to resolver B1.
4. Resolver B1 obtains a VR name for the IP address and returns the VR name to resolver A1.
5. Resolver A1 forwards the VR name to resolver C1.

6. Resolver C1 obtains an SAE reference for the VR and returns the SAE reference to resolver A1.
7. Resolver A1 passes the SAE reference to its host.
8. The host returns the SAE reference to the NIC proxy.

Figure 23 shows the interactions of the NIC components for this realm.

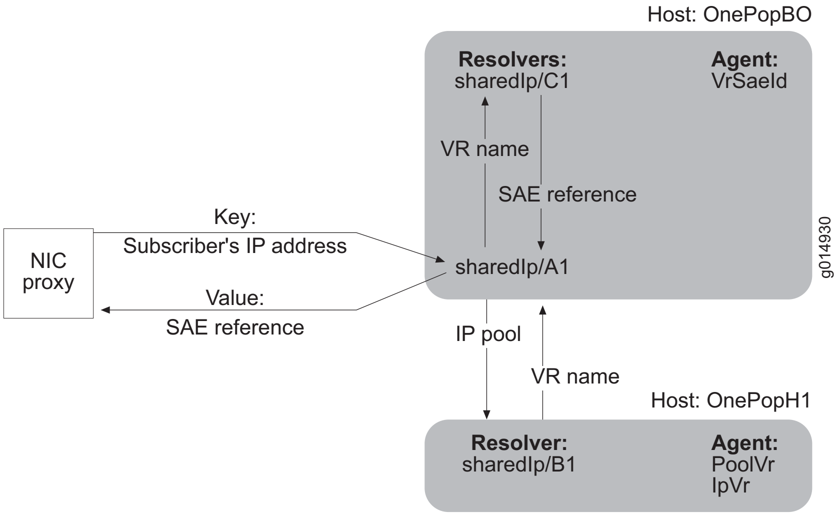
Figure 23: OnePopSharedIP Centralized Configuration



Distributed Configuration

In this configuration, the agents and resolvers are distributed among several hosts. When the NIC proxy sends a subscriber's IP address to the host OnePopBO, the resolvers execute the same actions as they do in the centralized configuration. Figure 24 illustrates the interactions of the NIC components for this realm.

Figure 24: OnePopSharedIP Distributed Configuration

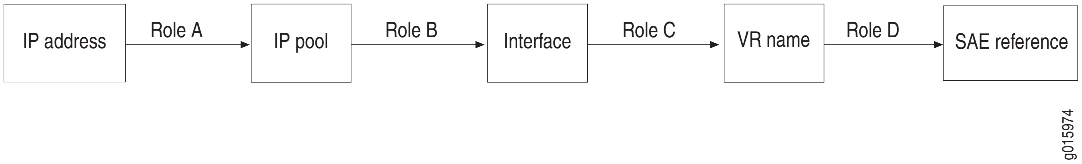


OnePopStaticRouteIp

The OnePopStaticRouteIp configuration scenario for NIC resolves an assigned IP address for a subscriber whose traffic enters the network through an interface on a JUNOS routing platform to a reference for the SAE that manages the interface. The realm for this configuration accommodates the situation in which the network publisher component gathers interface information for the JUNOS routing platforms. The resolution process takes a subscriber's IP address as a key and returns a reference to the SAE that manages the interface.

Figure 25 shows the resolution graph for this realm.

Figure 25: Resolution Process for the StaticRouteIp Realm



The following agents collect information for resolvers in this realm:

- Directory agent PoolInterface collects and publishes information about the mappings of IP address pools to interfaces.
- Directory agent VrSaeId collects and publishes information about the mappings of VRs to SAEs.

The agents obtain information from the interfaceConfiguration attribute of the EdgeRouter entry in the directory and read an XML document that conforms to the networkConfig.xsd schema. If this scenario is used with a different router type, you can edit the XML document.

For information about the XML document, see *Chapter 12, Obtaining Interface Configuration for OnePopStaticRouteIp on Solaris Platforms*.

The OnePopStaticRouteIp scenario provides two host configurations: a centralized configuration and a distributed configuration.

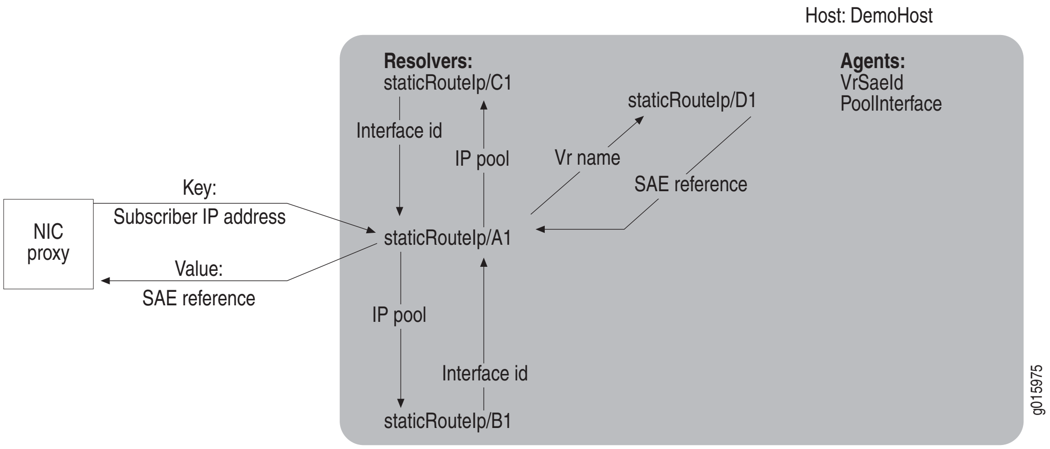
Centralized Configuration

In this configuration, the single host DemoHost supports all agents and resolvers. When the NIC proxy sends a subscriber's IP address to host DemoHost, the following sequence of events occurs:

1. The host passes the subscriber's IP address to resolver A1.
2. Resolver A1 obtains an IP pool for the IP address.
3. Resolver AI forwards the IP pool name to Resolver B1.
4. Resolver B1 obtains the interface ID for the IP pool and returns this value to resolver A1.
5. Resolver A1 forwards the interface ID to Resolver C1.
6. Resolver C1 resolves the interface ID to the VR name and returns the VR name to resolver A1.
7. Resolver A1 forwards the VR name to resolver D1.
8. Resolver D1 obtains a reference for the SAE managing the VR and returns the SAE reference to resolver A1.
9. Resolver A1 passes the SAE reference to its host.
10. The host returns the SAE reference to the NIC proxy.

Figure 26 shows the interactions of the NIC components for this realm.

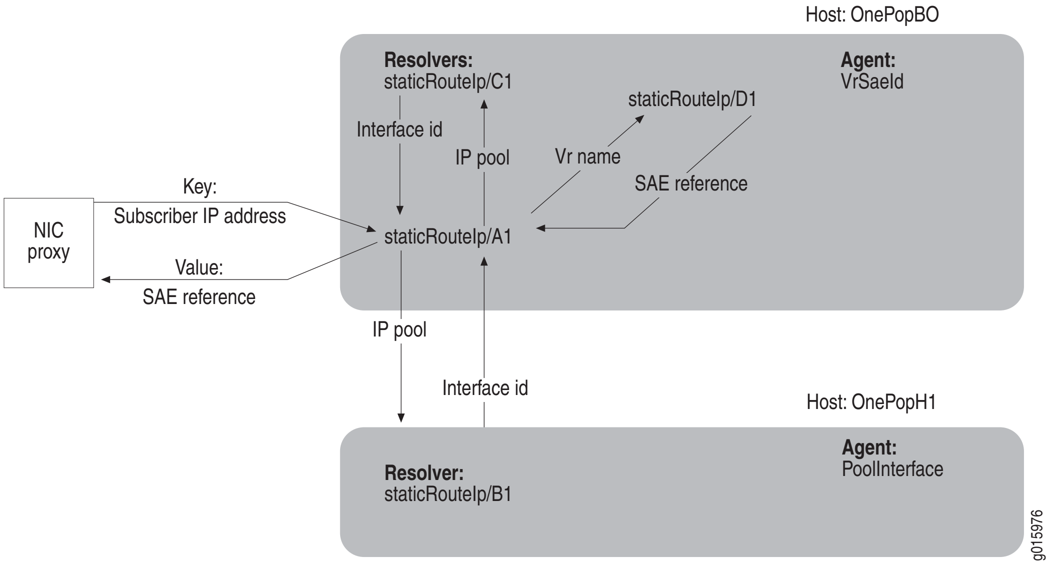
Figure 26: OnePopStaticRouteIp Centralized Configuration



Distributed Configuration

In this configuration, the agents and resolvers are distributed among two hosts. When a NIC proxy sends a subscriber IP address to host OnePopBO, the resolvers execute the same actions as they do in the centralized configuration. Figure 27 illustrates the interactions of the NIC components for this realm.

Figure 27: OnePopStaticRouteIp Distributed Configuration

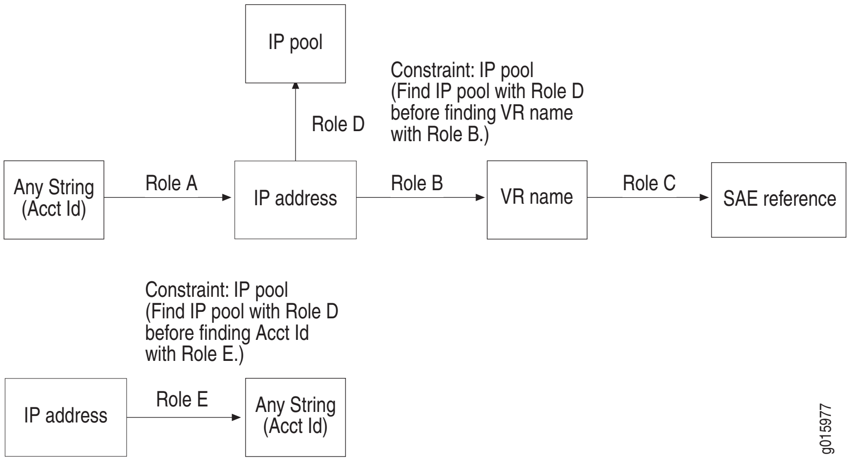


OnePopAcctId Scenario

This scenario illustrates a configuration in which subscribers have an accounting ID, as defined by the LDAP attribute accountingUserId or the plug-in attribute PA_ACCOUNTING_ID. The realms for this configuration accommodate two independent resolution processes, which can be used by the SRC Volume-Tracking Application (SRC-VTA).

Figure 28 shows the resolution graphs for this realm.

Figure 28: Resolution Process for acctId Realm



The following agents collect information for resolvers in this realm:

- Directory agent PoolVr collects and publishes information about the mappings of IP address pools to VRs.
- Directory agent VrSaeId collects and publishes information about the mappings of virtual routers and the mappings between virtual routers and SAEs.
- SAE plug-in agent AcctIdIp collects and publishes information about the mappings of accounting IDs of subscribers to subscriber IP addresses.
- SAE plug-in agent IpAcctId collects and publishes information about the mappings of subscriber IP addresses to accounting IDs.

The OnePopAcctId scenario provides one host for a centralized configuration. In this configuration the single host DemoHost supports all agents and resolvers. Two NIC proxies are associated with the configuration. One NIC proxy (called acct-sae in this description) submits accounting IDs, and another NIC proxy (called addr-acct in this description) submits subscribers’ IP addresses.

When the NIC proxy sends an accounting ID to host DemoHost, the following sequence of events occurs:

1. The host passes the subscriber’s accounting ID to resolver A1.
2. Resolver A1 obtains an IP address for the account ID.
3. Resolver A1 forwards the IP address to Resolver D1.
4. Resolver D1 obtains the IP pool for the IP address and returns it to Resolver A1.
5. Resolver A1 forwards the IP address and IP pool to Resolver B1.
6. Resolver B1 obtains the VR name and return it to resolve A1.

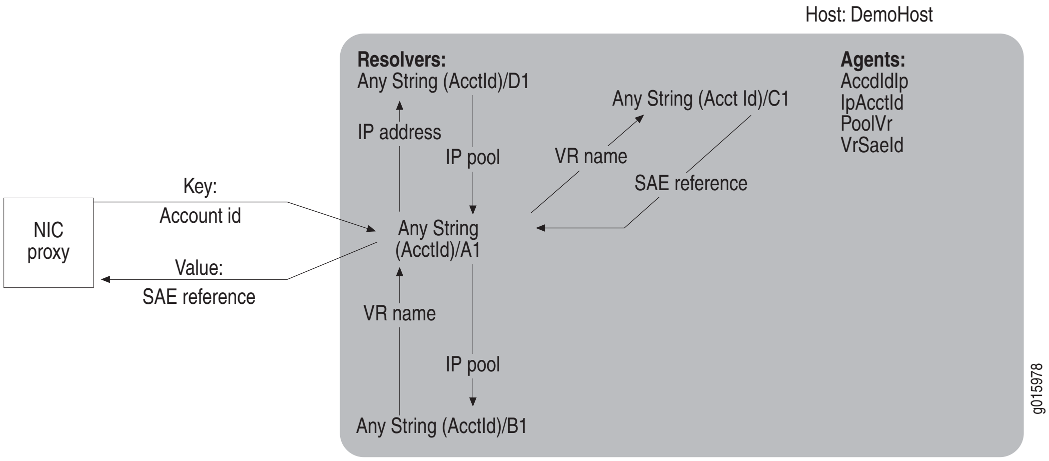
7. Resolver A1 forwards the VR name to resolver C1.
8. Resolver C1 obtains the SAE reference for the VR name and returns it to resolver A1.
9. Resolver A1 passes the SAE reference to its host.
10. The host returns the SAE reference to the NIC proxy acct-sae.

When the NIC proxy sends an IP address to host DemoHost, the following sequence of events occurs:

1. The host passes the subscriber's IP address to resolver A1.
2. Resolver A1 forwards the IP address to resolver D1.
3. Resolver D1 obtains the IP pool for the IP address and returns it to resolver A1.
4. Resolver A1 forwards the IP address and IP pool to resolver C1.
5. Resolver C1 obtains the accounting ID for the IP address and associated IP pool and returns the accounting Id to resolver A1.
6. Resolver A1 passes the accounting ID to its host.
7. The host returns the accounting ID to the NIC proxy addr-acct.

Figure 29 illustrates the interactions of the NIC components for this realm.

Figure 29: OnePopAcctId Centralized Configuration

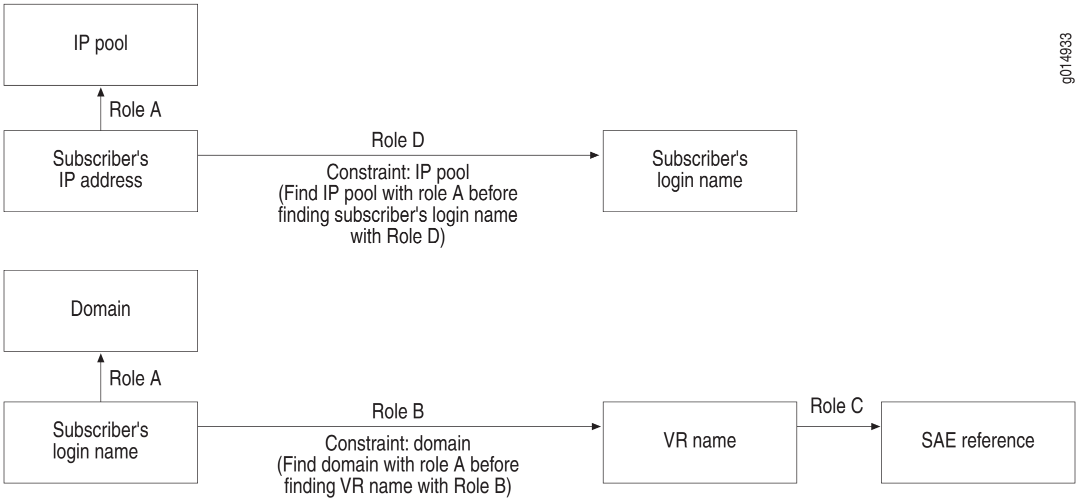


OnePopLogin Scenario

This scenario illustrates a configuration that is very similar to the OnePop scenario. The realm for this configuration accommodates two independent resolution processes, which are used by the SRC Volume Tracking Applications (SRC-VTAs) and may be used for other purposes.

Figure 30 shows the resolution graphs for this realm.

Figure 30: Resolution Processes login Realm


IP pool
Role A
Subscriber's IP address
Role D
Constraint: IP pool (Find IP pool with role A before finding subscriber's login name with Role D)
Subscriber's login name
Domain
Role A
Subscriber's login name
Role B
Constraint: domain (Find domain with role A before finding VR name with Role B)
VR name
Role C
SAE reference
g014933

The following agents interact with resolvers in this realm:

- SAE plug-in agent IpLoginName collects and publishes information about the mappings of IP addresses to login names.
- SAE plug-in agent LoginNameVr collects and publishes information about the mappings of login names to VRs.
- Directory agent Pool collects and publishes information about the IP address pools used by the VRs in a POP. The agent uses the information about the IP address pools to determine which resolver to communicate with, rather than communicating with all resolvers that are running role D.
- Directory agent VrSaeId collects and publishes information about the mappings of VRs to SAEs.

The OnePopLogin scenario provides two host configurations: a centralized configuration and a distributed configuration.

Centralized Configuration

In this configuration, single host DemoHost supports all agents and resolvers. Two NIC proxies are associated with this NIC configuration; one NIC proxy (called NIC proxy 1 in this documentation) submits subscribers' login names, and the other (called NIC proxy 2 in this documentation) submits subscribers' IP addresses.

When NIC proxy 1 sends a login name to the host DemoHost, the following sequence of events occurs:

1. The host passes the login name to resolver A1.
2. Resolver A1 obtains a domain name for the login name.

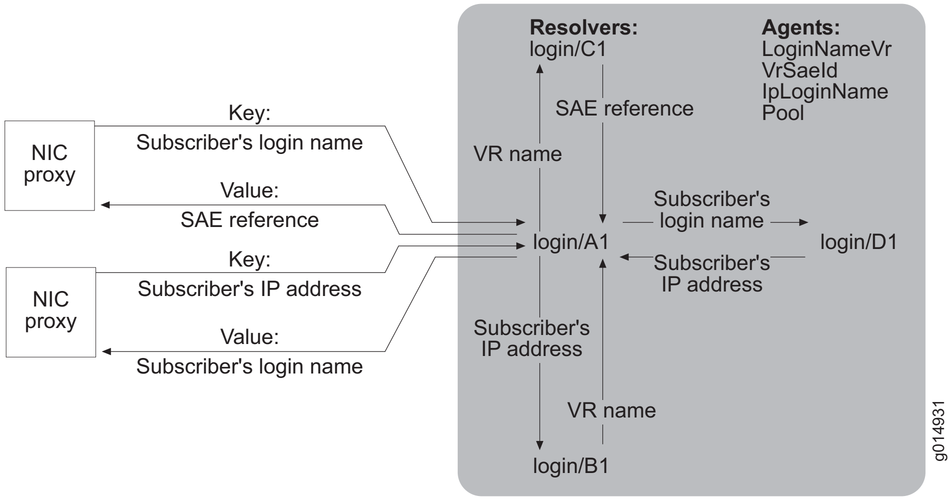
3. Resolver A1 forwards the login name and the domain to resolver B1.
4. Resolver B1 obtains a VR name for the login name and returns the VR name to resolver A1.
5. Resolver A1 forwards the VR name to resolver C1.
6. Resolver C1 obtains an SAE reference for the VR and returns the SAE reference to resolver A1.
7. Resolver A1 returns the SAE reference to its host.
8. The host returns the SAE reference to the NIC proxy.

When NIC proxy 2 sends a subscriber's IP address to host DemoHost, the following sequence of events occurs.

1. The host passes the IP address to resolver A1.
2. Resolver A1 obtains an IP pool for the IP address.
3. Resolver A1 forwards the IP address and the IP pool to resolver D1.
4. Resolver D1 obtains a login name for the IP address and returns the login name to resolver A1.
5. Resolver A1 passes the login name to its host.
6. The host returns the login name to the NIC proxy.

Figure 31 illustrates the interactions of the NIC components for this realm.

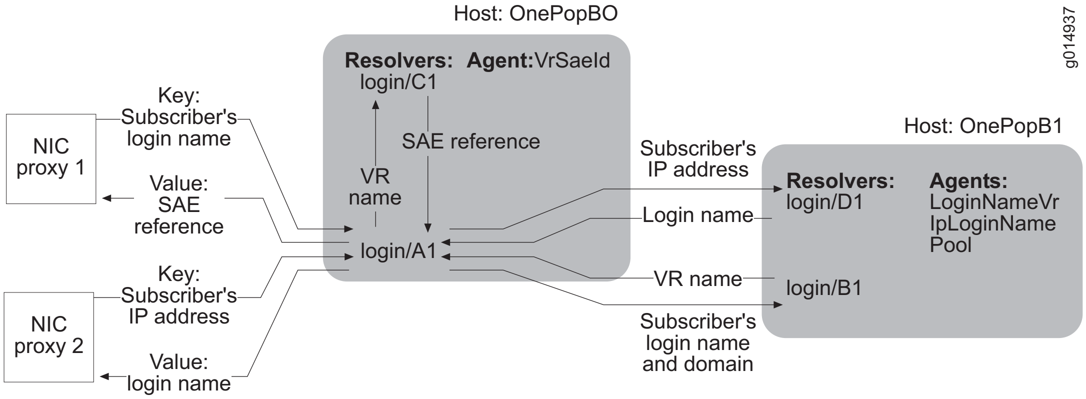
Figure 31: OnePopLogin Centralized Configuration



Distributed Configuration

In this configuration, the agents and resolvers are distributed among several hosts. When the NIC proxy sends a subscriber's IP address to the host OnePopBO, the resolvers execute the same actions as they do in the in the centralized configuration. Figure 32 illustrates the interactions of the NIC components for this realm.

Figure 32: OnePopLogin Distributed Configuration

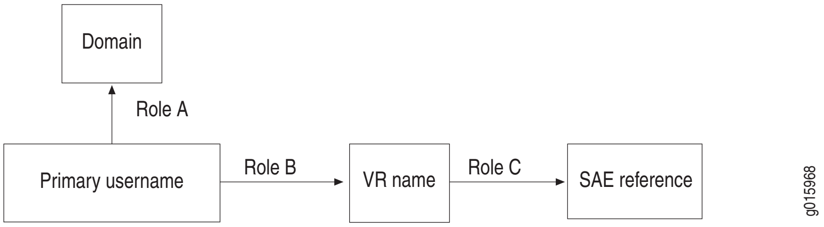


OnePopPrimaryUser

The OnePopPrimaryUser scenario is similar to one of the resolutions in the OnePopLogin scenario. In the OnePopPrimaryUser scenario, subscriber primary username, as identified by the PA_PRIMARY_USER_NAME attribute, is resolved to a reference for a managing SAE. The realm for this configuration accommodates a situation in which a NIC proxy provides a primary username.

Figure 33 show the resolution graph for this realm.

Figure 33: Resolution Processes for primary_user Realm



The following agents interact with resolvers in this realm:

- Directory agent VrSaeId collects and publishes information about virtual routers and the mappings between virtual routers and SAEs.
- SAE plug-in agent UserNameVr collects and publishes information about the mappings of subscriber primary usernames to VR names.

The OnePopPrimaryUser scenario provides two host configurations: a centralized configuration and a distributed configuration.

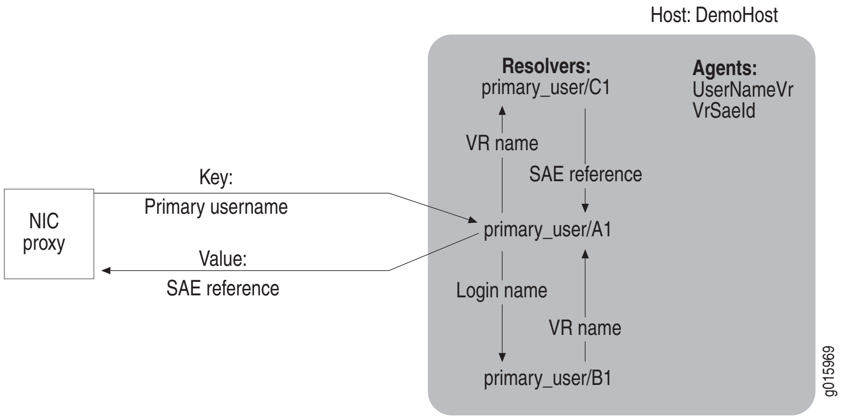
Centralized Configuration

In this configuration, a single host called DemoHost supports all agents and resolvers. When a NIC proxy send a subscriber's primary username to host Demo Host, the following sequence of events occurs:

1. The host passes the primary username to resolver A1.
2. (Optional) Resolver A1 resolves the primary username to its domain.
3. Resolver A1 forwards the primary username to resolver B1.
4. Resolver B1 obtains the name of the VR associated with the subscriber's primary username and returns the VR to resolver A1.
5. Resolver A1 forwards the VR to resolver C1.
6. Resolver C1 obtains the SAE reference for the SAE managing the VR and returns the SAE reference to resolver A1.
7. Resolver A1 returns the SAE reference to the host.
8. The host returns the SAE reference to the NIC proxy.

Figure 34 illustrates the interactions of the NIC components for this realm.

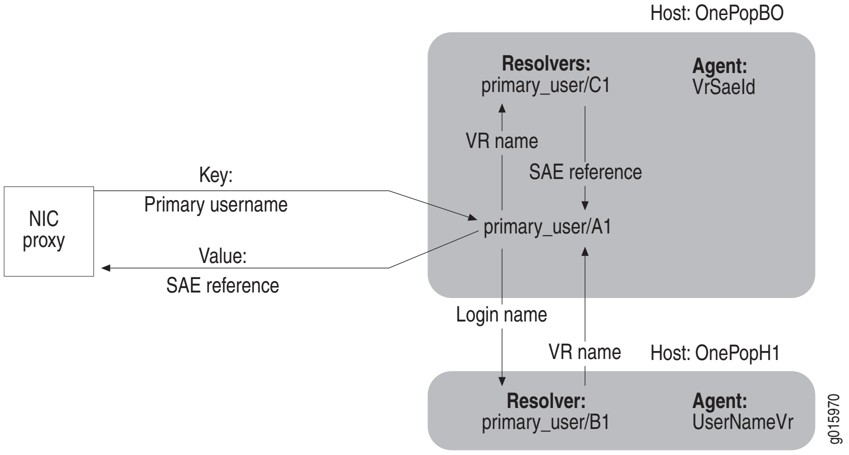
Figure 34: OnePopPrimaryUser Centralized Configuration



Distributed Configuration

In this configuration, the agents and resolvers are distributed among two hosts, When a NIC proxy sends a subscriber's primary username to the host OnePopBO, the resolvers execute the same actions as they do in the centralized configuration. Figure 35 illustrates the interactions of the NIC components for this realm.

Figure 35: OnePolpPrimaryUser Distributed Configuration



OnePopDnSharedIp Scenario

The OnePopDnSharedIp scenario illustrates how to configure SAE plug-in agents that have state synchronization enabled to support an SAE plug-in that uses state synchronization. This scenario uses the same centralized and distributed configurations of hosts as the OnePop scenario.

Two realms are configured:

- Shared IP

  The resolution process is identical to that for the OnePopShared scenario (see Figure 22 on page 246).

- DN realm

  This realm uses essentially the same resolution process as the MultiPop DN realm (see Figure 43 on page 269). However, some of the constraints differ.

  This realm also uses the same agents as the MultiPop DN realm. The names of agents and resolvers are essentially the same as those in the MultiPop configuration, although they do not include a POP identifier. Figure 36 on page 258 illustrates the centralized configuration, and Figure 37 on page 260 illustrates the distributed configuration for the DN realms.

The configuration for the two realms is similar to the configuration for the shared IP and DN realms in the OnePopAllRealms scenario. See *OnePopAllRealms Scenario* on page 261.

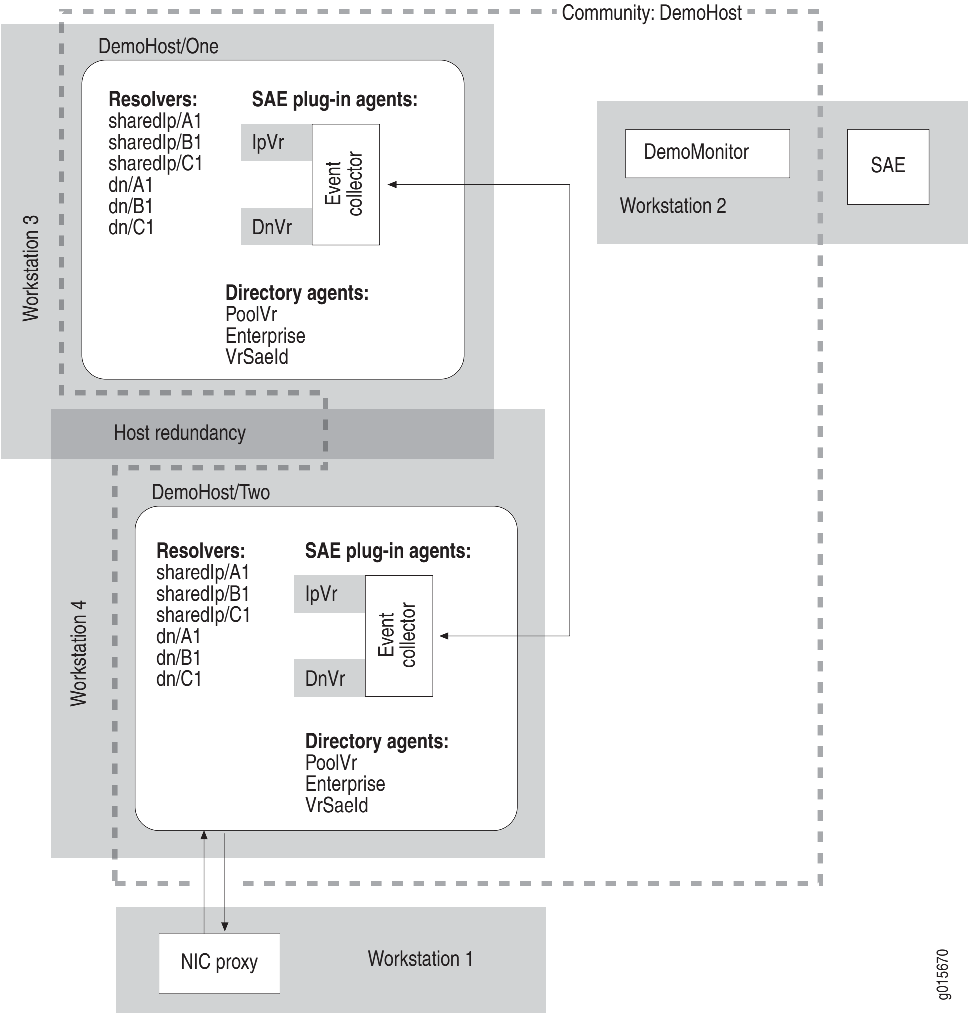
The OnePopAllRealms illustrates SAE plug-in agents configured to use SAE plug-in redundancy rather than SAE plug-in agents.

Centralized Configuration

Figure 36 on page 258 shows the centralized configuration for the scenario. Host DemoHost supports all resolvers and agents. The two SAE plug-in agents, IpVr and DnVr, share an event collector. Both plug-in agents have state synchronization enabled.

DemoHost is also configured for redundancy. Its redundant hosts (DemoHost/One and DemoHost/Two) perform the host function. The redundant hosts are on different machines, and both hosts support the resolvers and agents assigned to the parent host. The redundant hosts form a community called DemoHost with the monitor DemoMonitor, which tracks them.

Figure 36: OnePopDnSharedIp Realms Centralized Configuration



Distributed Configuration

Figure 37 on page 260 shows the distributed configuration from the scenario. Host OnePopBO supports two resolvers for each realm and a directory agent that is used by different realms. Host OnePopH1 supports one resolver for each realm and agents that are used by different realms.

Both hosts also have a redundant configuration. The redundant hosts for OnePopBO (OnePopBO/One and OnePopBO/Two) perform the host function. The redundant hosts are on different machines, and both hosts support the resolvers and agents assigned to the parent host.

The redundant hosts for OnePopH1 (OnePopH1/One and OnePopH1/Two) perform the host function. The redundant hosts are on different machines, and both hosts support the resolvers and agents assigned to the parent host.

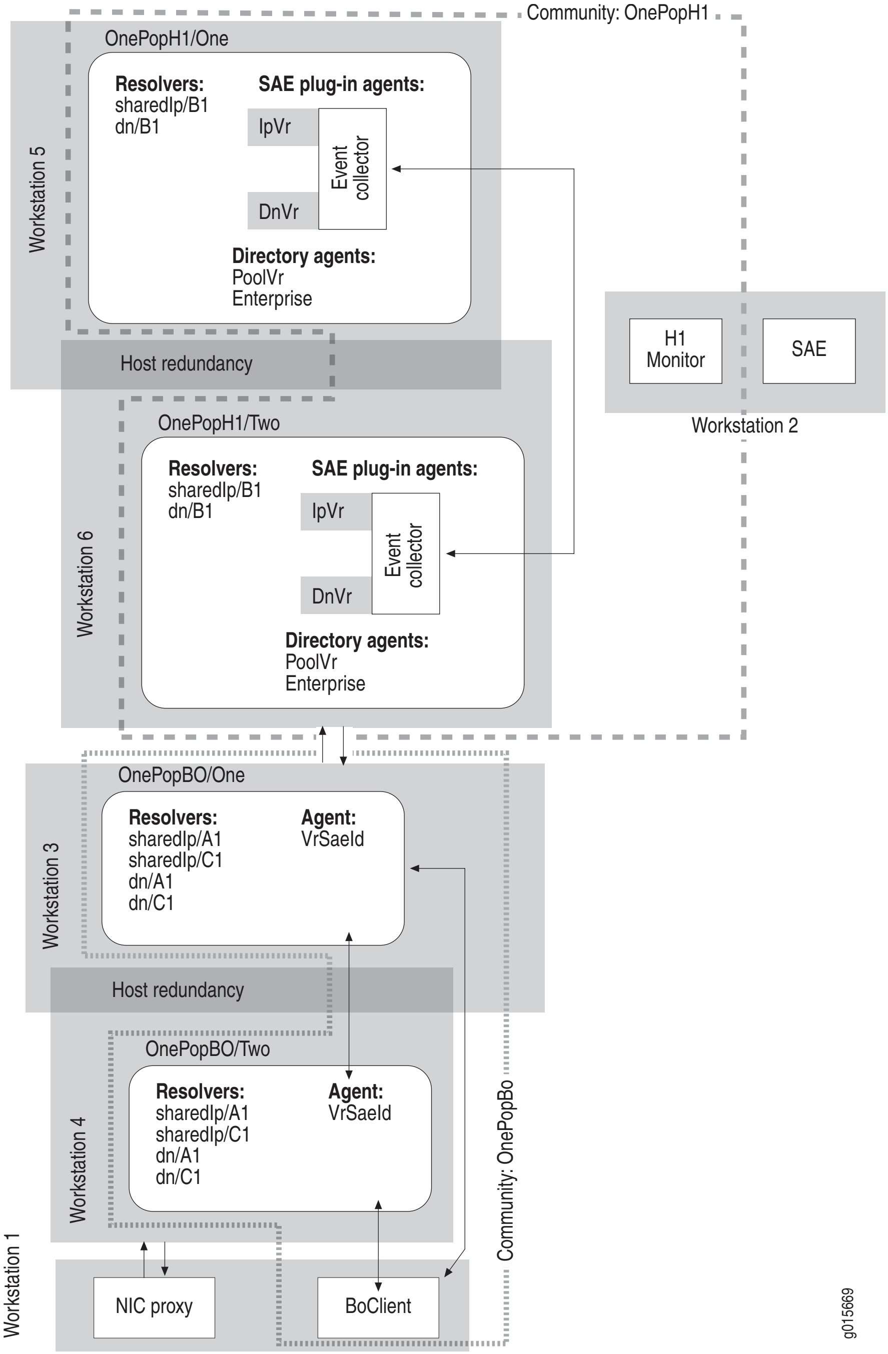
However, host OnePopH1 also supports two SAE plug-in agents, IpVr and DnVr, which share an event collector. These agents have state synchronization enabled.

The redundant hosts OnePopBO/One and OnePopBO/Two are members of a community called OnePopBO. This community supports the monitor, BoClient, which is installed on the machine that supports the NIC proxy. BoClient tracks the connections between the redundant hosts OnePopBO/One and OnePopBO/Two from the point of view of the NIC client (NIC proxy).

Similarly, the redundant hosts OnePopH1/One and OnePopH1/Two are members of a community called OnePopH1. This community has one monitor, H1Monitor, which is located on the same machine as the SAE and tracks the connections among the redundant hosts in the same community, their primary host, and the other hosts in the configuration.

H1Monitor comprises the monitor process OnePop, which is installed on the same machine as the SAE. BoClient comprises the monitor process OnePopClient, which is installed on the same machine as the NIC proxy.

Figure 37: OnePopDnSharedIp Realms Distributed Configuration



OnePopAllRealms Scenario

The main purpose of the OnePopAllRealms scenario is to illustrate how to configure redundancy. This scenario uses the same centralized and distributed configurations of hosts as the OnePop scenario.

Three realms are configured:

- IP realm

  This realm uses essentially the same resolution process as the IP realm for the OnePop scenario (see Figure 12 on page 238). However, some of the constraints differ.

- Shared IP

  The resolution process is identical to that for the OnePopShared scenario (see Figure 22 on page 246).

- DN realm

  This realm uses essentially the same resolution process as the MultiPop DN realm (see Figure 43 on page 269). However, some of the constraints differ.

  This realm also uses the same agents as the MultiPop DN realm. The names of agents and resolvers are essentially the same as those in the MultiPop configuration, although they do not include a POP identifier. By reviewing the scenario, Figure 38 and Figure 39, you can determine exact pictures of the DN realms for the centralized and distributed configurations.

Centralized Configuration

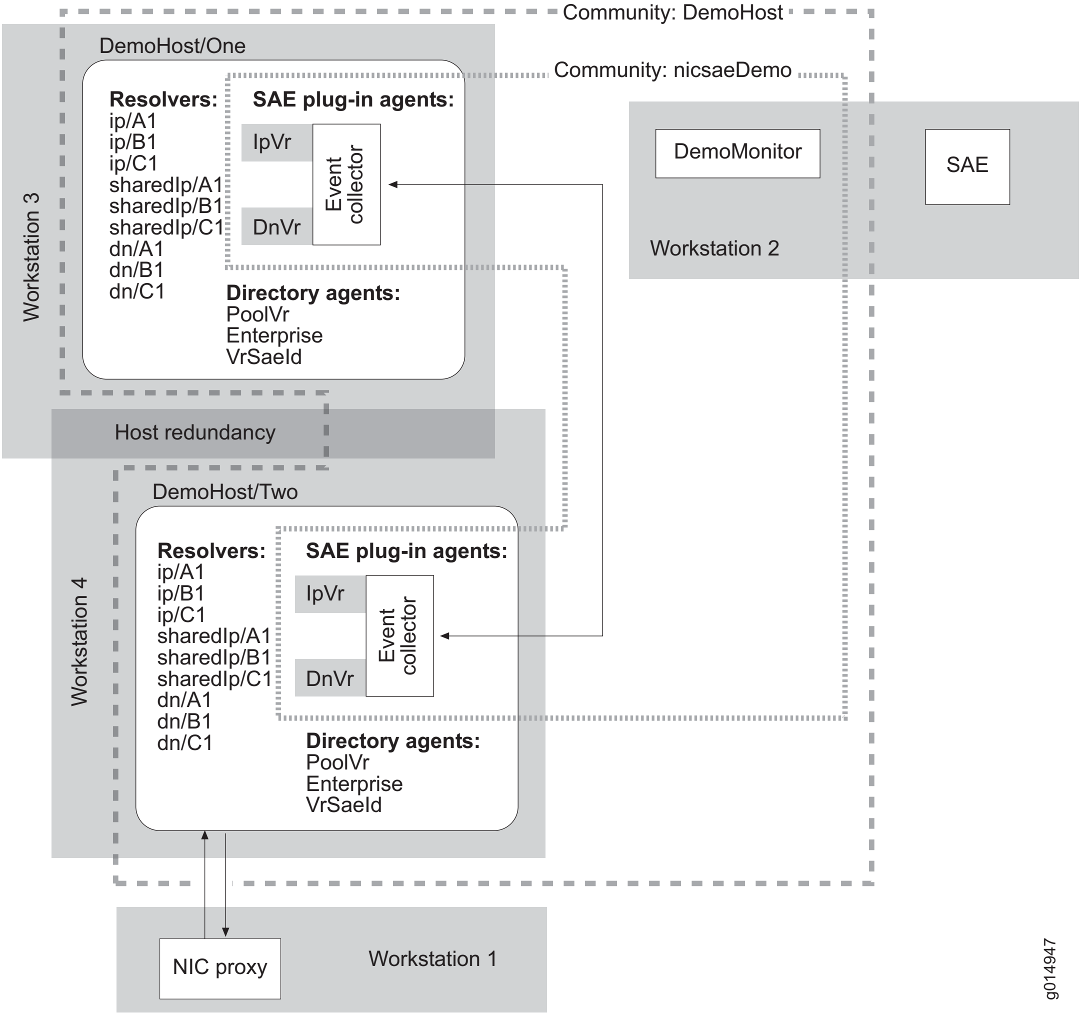
Figure 38 on page 262 shows the centralized configuration for the scenario. Host DemoHost supports all resolvers and agents. However, because host DemoHost is configured for redundancy, its redundant hosts (DemoHost/One and DemoHost/Two) perform the host function. The redundant hosts are on different machines, and both hosts support the resolvers and agents assigned to the parent host.

The parent host DemoHost also supports two SAE plug-in agents, IpVr and DnVr, which share an event collector. Each SAE plug-in agent has a redundant agent called Demo; these redundant agents also share an event collector. The redundant agents and their shared event collector are assigned to both redundant hosts DemoHost/One and DemoHost/Two.

The redundant agents form a community called nicsaeDemo with the monitor DemoMonitor, which tracks them. The redundant agents are identified in the community by the names DemoHost/One and DemoHost/Two; these names specify their hosts and provide unique identifiers for the redundant agents.

The redundant hosts form a community called DemoHost with the monitor DemoMonitor, which tracks them.

Figure 38: OnePopAllRealms Centralized Configuration



Distributed Configuration

Figure 39 on page 264 shows the distributed configuration for the scenario. Host OnePopBO supports two resolvers for each realm and a directory agent that is used by different realms. However, because host OnePopBO is configured for redundancy, its redundant hosts (OnePopBO/One and OnePopBO/Two) perform the host function. The redundant hosts are on different machines, and both hosts support the resolvers and agents assigned to the parent host.

Host OnePopH1 supports one resolver for each realm and agents that are used by different realms. Host OnePopH1 is also configured for redundancy, and its redundant hosts (OnePopH1/One and OnePopH1/Two) perform the host function. The redundant hosts are on different machines, and both hosts support the resolvers and agents assigned to the parent host.

However, host OnePopH1 also supports two SAE plug-in agents, IpVr and DnVr, which share an event collector. Each SAE plug-in agent has a redundant agent called onePop; these redundant agents also share an event collector. The redundant agents and their shared event collector are assigned to redundant hosts OnePopH1/One and OnePopH1/Two.

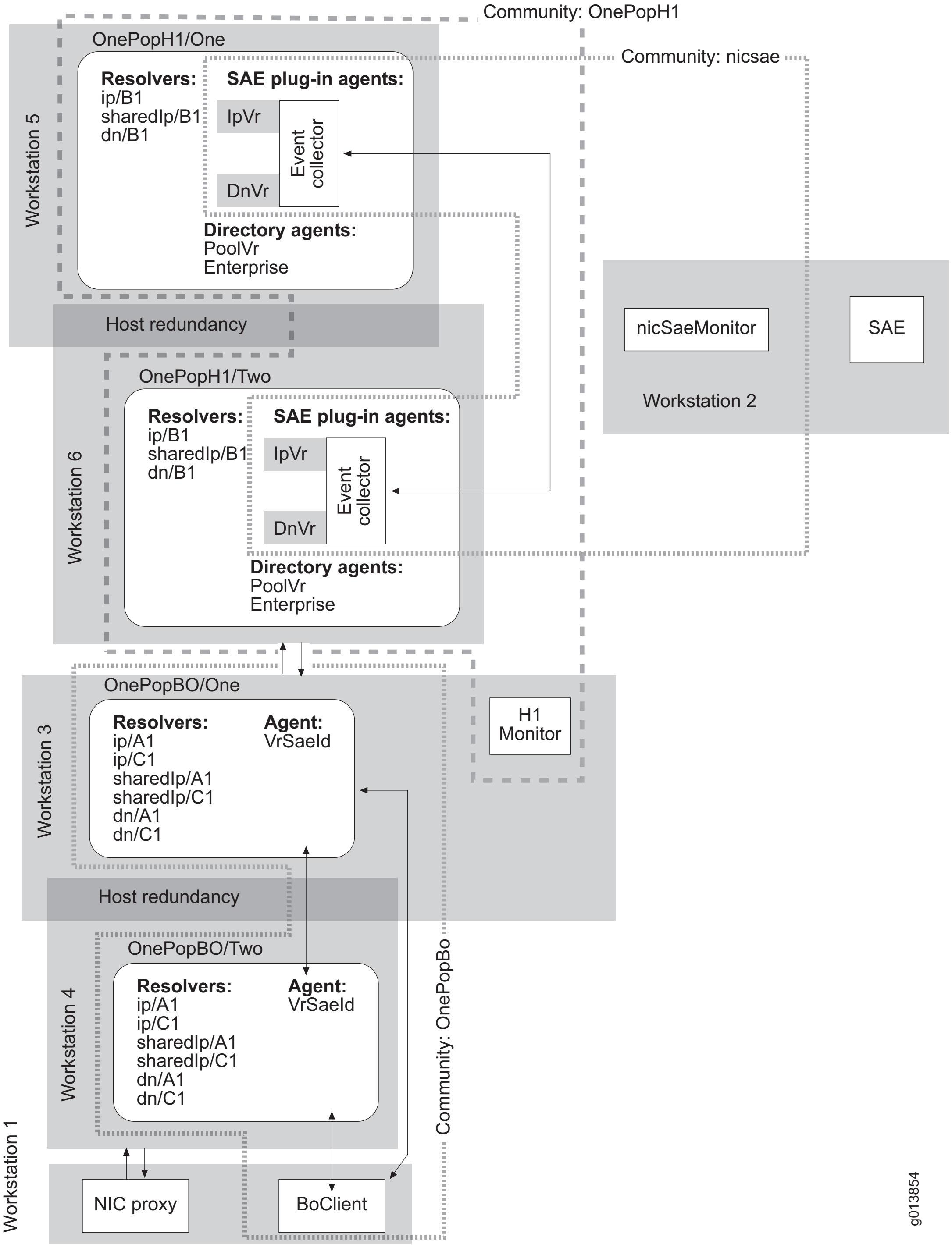
The redundant agents form a community called nicsae with monitor nicSaeMonitor, which tracks them. The redundant agents are identified in the community by the names OnePopH1/One and OnePopH1/Two; these names specify their hosts and provide unique identifiers for the redundant agents.

The redundant hosts OnePopBO/One and OnePopBO/Two are members of a community called OnePopBO. This community supports the monitor, BoClient, which is installed on the machine that supports the NIC proxy. BoClient tracks the connections between the redundant hosts OnePopBO/One and OnePopBO/Two from the point of view of the NIC client (NIC proxy).

Similarly, the redundant hosts OnePopH1/One and OnePopH1/Two are members of a community called OnePopH1. This community has one monitor, H1Monitor, which is located on the same machine as the SAE and tracks the connections among the redundant hosts in the same community, their primary host, and the other hosts in the configuration.

H1Monitor and nicSaeMonitor are part of the monitor process OnePop, which is also installed on the same machine as the SAE. BoClient is part of the monitor process OnePopClient, which is installed on the same machine as the NIC proxy.

Figure 39: OnePopAllRealms Distributed Configuration



MultiPop Scenario

The MultiPop scenario illustrates a configuration that involves two POPs: Montreal and Ottawa. This configuration does not provide redundancy. The NIC proxy communicates with the back office host (BackOffice), which in turn communicates with the POP hosts (MontrealHost and OttawaHost). Hosts MontrealHost and OttawaHost support equivalent hosts and agents and manage resolutions in the same way.

When host BackOffice receives a data key from the NIC proxy, the following sequence of events occurs:

1. Host BackOffice forwards requests as follows:
    - If the request is for the Montreal POP, host BackOffice forwards the request to POP host MontrealHost.
    - If the request is for the Ottawa POP, host BackOffice forwards the request to POP host OttawaHost.
2. Delegating tasks to other resolvers as necessary, the resolvers in the POP obtain data values that correspond to the data key request, and return them.
3. The POP host returns the data values to host BackOffice, which returns the value to the NIC proxy.

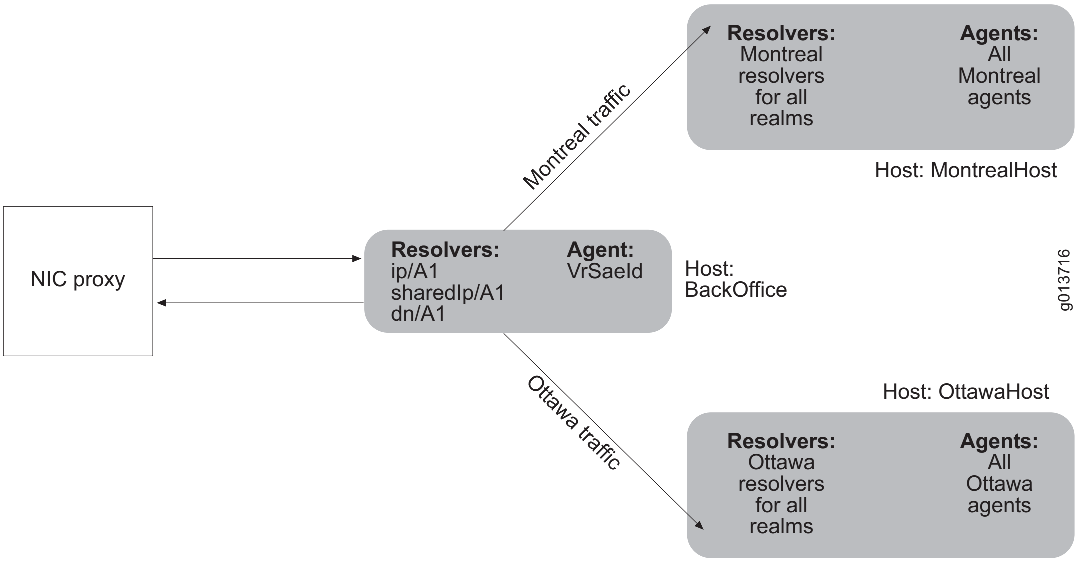
The scenario shows three realms for this configuration:

- IP
- Shared IP
- DN

Each realm provides a different type of resolution. The following sections provide information about these realms.

Figure 40 illustrates this configuration.

Figure 40: MultiPop Configuration



IP Realm

This realm accommodates the situation in which IP address pools are configured locally on each VR. 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. This realm uses essentially the same resolution process as the ip realm for the OnePop scenario (see Figure 12 on page 238). However, some of the constraints differ.

The following agents interact with the resolvers in this realm:

- Directory agents montrealPoolVr and ottawaPoolVr collect and publish information that maps IP address pools to VRs. Each agent publishes only the information that is relevant to its POP. You achieve selective publishing by relating an Ottawa scope to the VRs in the Ottawa POP and a Montreal scope to the VRs in the Montreal POP and defining a search filter for the agents to load only the VRs in its POP.
- Directory agent VrSaeId in the back office collects and publishes information that maps VRs to SAEs for both POPs.

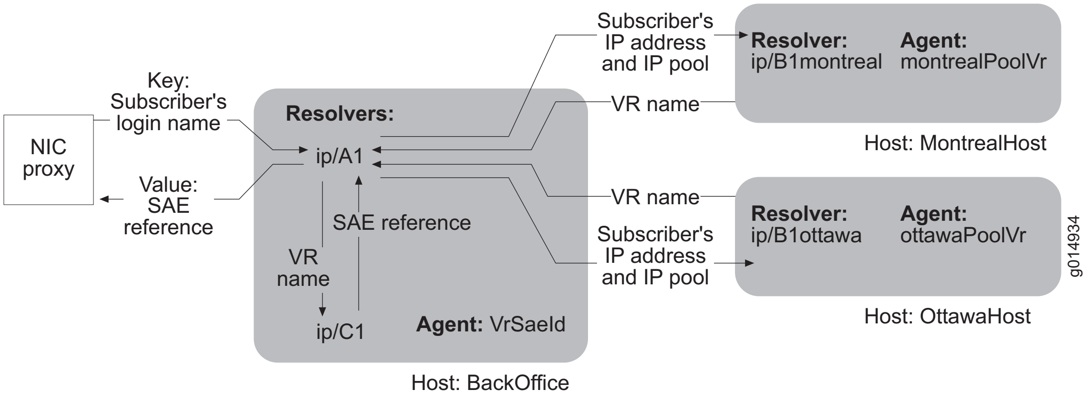
When the NIC proxy sends a subscriber's IP address to host BackOffice, the following sequence of events occurs:

1. Host BackOffice passes the IP address to resolver ip/A1.
2. Resolver ip/A1 obtains an IP pool for the IP address.
3. Resolver ip/A1, based on the value of the IpPool, forwards the request to ip/B1montreal or ip/B1ottawa.

4. Resolver ip/B1montreal or resolver ip/B1ottawa obtains a VR name for this IP pool and returns the VR name to resolver ip/A1.
5. Resolver ip/A1 forwards the VR name to resolver ip/C1.
6. Resolver ip/C1 obtains the SAE identity for this VR and returns the value to resolver ip/A1.
7. Resolver ip/A1 returns the SAE reference to its host.
8. Host BackOffice returns the SAE reference to the NIC proxy.

Figure 41 illustrates the interactions of the NIC components for this realm.

Figure 41: iP Realm for MultiPop Configuration



Shared IP Realm

This realm accommodates the situation in which IP address pools are shared by VRs in the same POP. The realm takes a subscriber’s IP address as the key and returns the corresponding SAE as the value. Figure 13 on page 239 shows the resolution graph for this realm.

The following agents interact with resolvers in this realm:

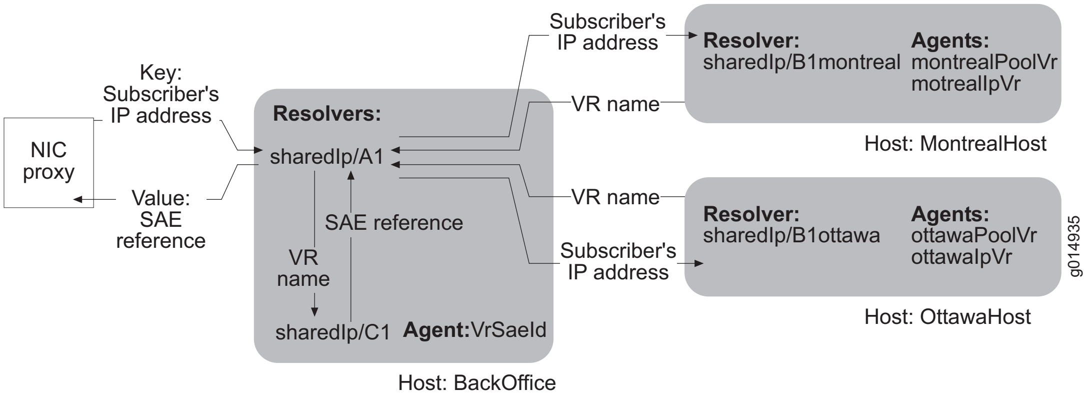
- Directory agents montrealPoolVr and ottawaPoolVr collect and publish information about the mappings of IP address pools to VRs. Each agent publishes only the information that is relevant to its POP.
- SAE plug-in agents montrealIpVr and ottawaIpVr collect and publish information about the mappings of subscriber IP addresses to VRs. Each agent publishes only the information that is relevant to its POP.
- Directory agent VrSaeId in the back office collects and publishes information about the mappings of VRs to SAEs for both POPs.

When the NIC proxy sends a subscriber's IP address to host BackOffice, the following sequence of events occurs:

1. Host BackOffice passes the IP address to resolver sharedIp/A1.
2. Resolver sharedIp/A1 obtains an IP pool for the IP address.
3. Resolver sharedIp/A1, based on the value of the IP pool, forwards the request to sharedIp/B1montreal or sharedIp/B1ottawa.
4. Resolver sharedIp/B1montreal or resolver sharedIp/B1ottawa obtains a VR name for this IP address and returns the VR name to resolver sharedIp/A1.
5. Resolver sharedIp/A1 forwards the VR name to resolver sharedIp/C1.
6. Resolver sharedIp/C1 obtains the SAE identity for this VR and returns the value to resolver sharedIp/A1.
7. Resolver sharedIp/A1 passes the SAE reference to its host.
8. Host BackOffice returns the SAE reference to the NIC proxy.

Figure 42 illustrates the interactions of the NIC components for this realm.

Figure 42: sharedIP Realm for MultiPop Configuration

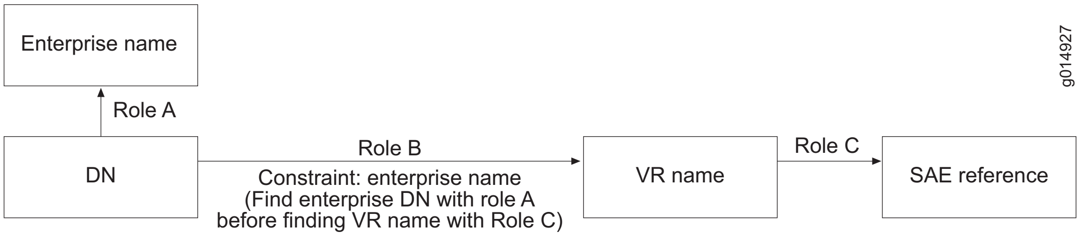


DN Realm

The DN realm takes the DN of an access subscriber (an access DN) as the key and returns the corresponding SAE as the value. Figure 43 shows the resolution process for this realm.

Figure 43 shows the resolution graph for this realm.

Figure 43: Resolution Graph for MultiPOP dn Realm


Enterprise name
Role A
DN
Role B
Constraint: enterprise name
(Find enterprise DN with role A
before finding VR name with Role C)
VR name
Role C
SAE reference
g014927

The following agents interact with resolvers in this realm:

- Directory agents ottawaEnterprise and montrealEnterprise collect and publish information about the DNs of enterprise subscribers (enterprise DNs). Each agent publishes only the information that is relevant to its POP. You achieve selective publishing by relating an Ottawa service scope to the enterprises in the Ottawa POP and a Montreal service scope to the enterprises in the Montreal POP and defining a search filter for the agents to load only the enterprises in its POP.
- SAE plug-in agents montrealDnVr and ottawaDnVr collect and publish information about the mappings of access DNs to VRs. Each agent publishes only the information that is relevant to its POP.
- Directory agent VrSaeId collects and publishes information about the mappings of VRs to SAEs for both POPs.

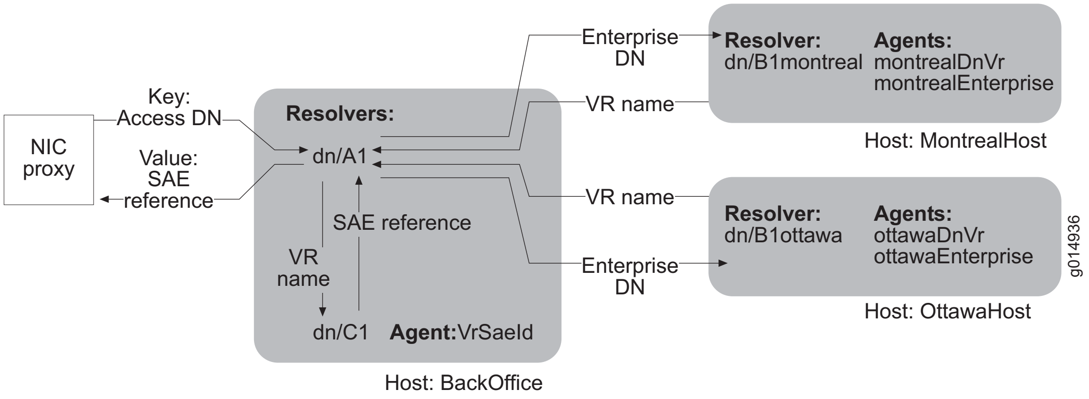
When the NIC proxy sends an access DN to host BackOffice, the following sequence of events occurs:

1. Host BackOffice passes the access DN to resolver dn/A1.
2. Resolver dn/A1 obtains an enterprise DN for the access DN.
3. Resolver dn/A1, based on the value of the enterprise DN, forwards the request to dn/B1montreal or dn/B1ottawa.
4. Resolver dn/B1montreal or resolver dn/B1ottawa obtains a VR name for this enterprise DN and returns the VR name to resolver dn/A1.
5. Resolver dn/A1 forwards the VR name to resolver dn/C1.
6. Resolver dn/C1 obtains the SAE reference for this VR and returns the value to resolver dn/A1.

7. Resolver dn/A1 passes the SAE reference to its host.
8. Host BackOffice returns the SAE reference to the NIC proxy.

Figure 44 illustrates the interactions of the NIC components for this realm.

Figure 44: dn Realm for MultiPop Configuration



Part 4

Providing Admission Control with SRC-ACP

Chapter 20

Overview of Providing Admission Control with SRC-ACP

This chapter describes the SRC Admission Control Plug-In (SRC-ACP) application, which provides admission control. Topics include:

Overview of SRC-ACP

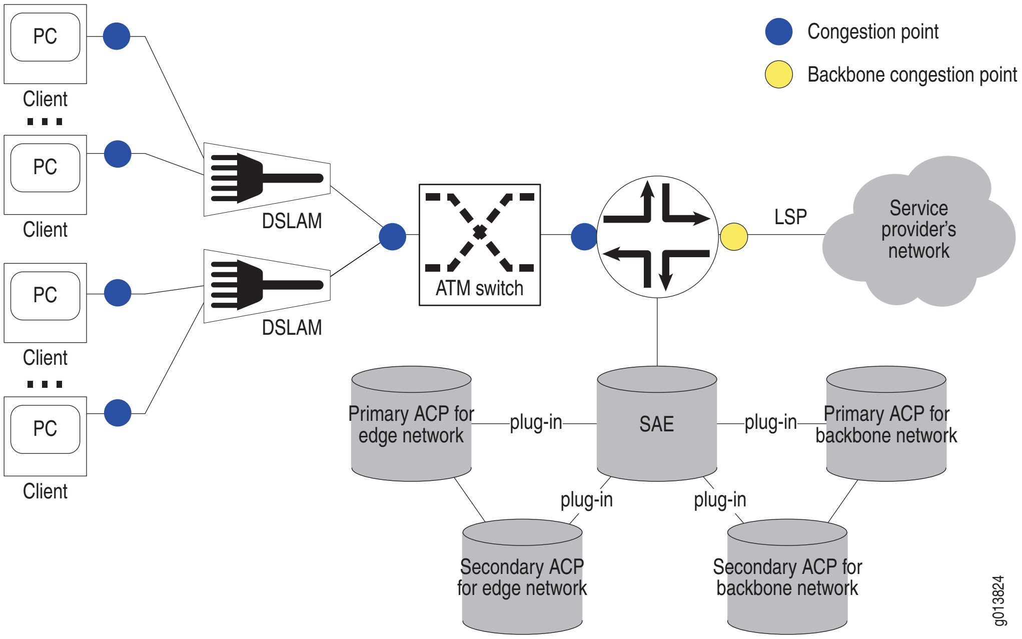
SRC-ACP is an external plug-in for the SAE. SRC-ACP authorizes and tracks subscribers' use of network resources associated with services that the SRC software manages. Service providers can implement SRC-ACP configurations for both residential and enterprise subscribers. Consequently, both JUNOSe routers and JUNOS routing platforms are compatible with SRC-ACP. References to virtual routers (VRs) in this documentation refer to an actual VR on a JUNOSe router or the single VR called default that the SRC software associates with each JUNOS routing platform (see *Part 2*, *Using Juniper Networks Routers in the SRC Network*).

SRC-ACP operates in two separate regions of the SRC network: the *edge* network and the *backbone* network. The edge network is the layer 2 access network through which subscribers connect to the router. The backbone network is the region between the router and the service provider's network.

Congestion often occurs in the network at points where connections are aggregated. SRC-ACP monitors congestion points at interfaces between devices in the edge network. In the backbone network, SRC-ACP monitors one congestion point, a point-to-point label-switched path (LSP) between the router and the service provider's network.

Figure 45 shows a typical network topology.

Figure 45: Position of SRC-ACP in Network


PC
Client
PC
Client
PC
Client
PC
Client
DSLAM
DSLAM
ATM switch
LSP
Service provider's network
Congestion point
Backbone congestion point
Primary ACP for edge network
plug-in
SAE
plug-in
Primary ACP for backbone network
plug-in
plug-in
Secondary ACP for edge network
Secondary ACP for backbone network
g013824

In the edge network, SRC-ACP performs the following procedures to determine whether there are sufficient resources to activate a service:

- Tracks active services for each subscriber and the guaranteed traffic rate (bandwidth) at the congestion points associated with a subscriber.
- Tracks the rate of traffic between the subscriber and the network (upstream bandwidth) and the rate of traffic between the network and subscriber (downstream bandwidth).
- Monitors new requests for activation of services.
- Compares the resources required for the new services with the resources available for the subscriber and the congestion points.
- Activates the service if sufficient resources are available, and prevents activation of the service if sufficient resources are not available.

In the backbone network, SRC-ACP performs the following procedures to determine whether there are sufficient resources to activate a service:

- Tracks the guaranteed traffic rate for a service at the congestion point.
- Tracks the actual traffic rate for the service at the congestion point.

- Monitors new requests for activation of services.
- Compares the resources required for the new services with the resources available at the congestion point.
- Activates the service if sufficient resources are available, and prevents activation of the service if sufficient resources are not available.

Typically, network administrators use their own network management applications and external applications to provide data for SRC-ACP. SRC-ACP first obtains updates from external applications through its remote CORBA interface, and then obtains updates from the directory by means of LDAP. For information about developing external applications that send data to SRC-ACP, see *API for ACP* on page 281. SRC-ACP does not interact directly with the network to assess the capacity of a congestion point or actual use of network resources.

In the backbone network, SRC-ACP can also execute applications defined in the action congestion point. For information about defining applications in congestion points, see *Configuring Action Congestion Points* on page 311. Some applications require real-time congestion point status. If SRC-ACP must provide real-time congestion point status to the application, state synchronization must be enabled to handle interface tracking events so that the congestion points are updated properly.

Deriving Congestion Points Automatically

SRC-ACP can derive some congestion points automatically. Depending on your network configuration and requirements, however, you may need to enter congestion points manually. This section describes the conditions and requirements for SRC-ACP to derive congestion points automatically.

Deriving Edge Congestion Points

For SRC-ACP to derive edge congestion points, subscribers must always connect through the same interface on the router. In addition, SRC-ACP requires one of the following conditions to derive edge congestion points if you are not using a congestion point profile:

- Access to subscriber profiles that define bandwidth values and a list of the distinguished names (DNs) of congestion points between the subscriber and the router.
- An ATM access network between the subscriber and the router for which all the traffic coming from one DSLAM travels on a single virtual path. In this case, SRC-ACP automatically derives three congestion points through the network access server (NAS) port ID. (For information about the NAS port ID, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 11, Configuring Accounting and Authentication Plug-Ins with the SRC CLI*.) Table 17 on page 276 shows the edge congestion points and the corresponding locations in the directory.

For more information about automatically deriving congestion points from a configured congestion point profile, see *Deriving Congestion Points from a Profile* on page 276.

SRC-ACP does not use bandwidth statistics from subscriber profiles when it derives congestion points, because the congestion points already use that data.

Table 17: Congestion Points Derived Through NAS Port ID

Congestion Points	Location of Object in Directory
Physical interface on router	interfaceName = ATM < slot > / < port > , *orderedCimKeys = < routerName > , o = AdmissionControl, o = umc* < slot > —Number of port on router < port > —Number of port on router < routerName > —Hostname configured for router
ATM virtual path	interfaceName = ATM < slot > / < port > : < vpi > *orderedCimKeys = < routerName > , o = AdmissionControl, o = umc* < vpi > —Number of virtual path on router
ATM virtual connection	interfaceName = ATM < slot > / < port > : < vpi > . < vci > *orderedCimKeys = < routerName > , o = AdmissionControl, o = umc* < vci > —Number of virtual connection on router

Deriving Congestion Points from a Profile

If you configure a congestion point profile, SRC-ACP can automatically derive congestion points for cases in which:

- There is no subscriber profile.
- The congestion points can be derived from information provided by the access interface on B-RAS. For example, in an ATM or VLAN connection, you can derive congestion points representing physical interfaces and intermediate switches based on the NAS port ID reported by B-RAS.

When SRC-ACP receives notification to start subscriber tracking and to load congestion points for a subscriber, it runs a congestion point classification and accesses the configured congestion point profile. Congestion point classification uses the same classification engine as subscriber and interface classification in the SAE.

For this feature to operate correctly, you create a congestion point profile that automatically performs congestion point classification. For more information about this topic, see *Defining a Congestion Point Profile* on page 321 (using the SRC CLI) or *Configuring SRC-ACP* on page 358 (using SRC configuration applications on Solaris platforms).

Deriving Backbone Congestion Points

SRC-ACP can automatically derive backbone congestion points if you specify the setting <-vrName->/<-serviceName-> for the congestion point associated with a service. When the SRC-ACP starts operating, it will substitute the name of the VR and the service name from the activation request.

For example, you can specify the setting <-vrName->/<-serviceName-> for the congestion point associated with a service called News. Then, when a subscriber who connects to the network through a VR called boston requests activation of this service, SRC-ACP receives the request and proceeds as follows:

1. SRC-ACP reads the congestion point specification, <-vrName->/<-serviceName->, from the congestion point defined for the service News.
2. SRC-ACP substitutes the actual information, boston/News, in the variables.
3. SRC-ACP uses this information to generate the DN *cn=News, cn=boston, o=CongestionPoints, o=umc*.
4. SRC-ACP uses this DN to obtain from the directory the network interface, which defines the location of the congestion point, for this DN.

For this feature to operate correctly, you must configure the DN for each combination of VR and service to point to an actual network interface. For more information about this topic, see *Configuring SRC-ACP to Manage the Backbone Network* on page 310.

Allocating Bandwidth to Applications Not Controlled by SRC-ACP

If you control the bandwidth of some applications by means of SRC-ACP, you can accommodate the applications that are not controlled by SRC-ACP by assigning *background* bandwidths for the edge congestion points. The background bandwidth is the total bandwidth allocated to the applications for which bandwidth is not controlled by SRC-ACP.

Because the total background bandwidth is unlikely to be used at a particular time, you can also specify a tuning factor that provides an estimation of the fraction of the background bandwidth that will be used. You can configure multiple values for the background bandwidth with corresponding tuning factors.

Use of Multiple SRC-ACPs

An SRC-ACP can support one or more SAEs. Similarly, multiple SRC-ACPs can support one SAE; for example, if an SAE is managing multiple VRs, you may have an SRC-ACP for each VR. However, only one SRC-ACP can manage a particular congestion point.

Interactions Between SRC-ACP and Other Components

This section describes how SRC-ACP interacts with other components to track data.

1. (Edge and dual mode only) When a subscriber connects to the router, SRC-ACP loads the subscriber profile from the directory. If the subscriber profile contains provisioned and actual traffic rates for the subscriber's interface and the set of congestion points between the subscriber and the router, SRC-ACP caches the information while the subscriber is connected to the router. SRC-ACP automatically updates the subscriber's actual upstream and downstream rates if the subscriber profile changes in the directory.

2. (Backbone mode only) When a subscriber activates a service, SRC-ACP loads the network interfaces defined in the service and caches the information.

3. (Optional) SRC-ACP obtains through its remote CORBA interface data from external applications about subscribers and congestion points. If a congestion point is unavailable, SRC-ACP denies service activation requests on the associated network interface until the interface is available again.

4. If SRC-ACP does not receive data from an external application, SRC-ACP loads data about congestion points from the directory. For each congestion point the following data is retrieved:

   - Provisioned bandwidth

   - Background bandwidths (if used for edge congestion points)

   SRC-ACP caches this information and automatically updates the cache when the information changes in the directory.

5. (Edge and dual modes) If SRC-ACP does not receive data from an external application, SRC-ACP loads a subscriber's provisioned or actual bandwidth from the subscriber profile. If the actual bandwidth is available, SRC-ACP ignores the provisioned bandwidth.

   SRC-ACP caches this information and automatically updates the cache when the information changes in the directory.

6. (Backbone and dual modes only) Using a hosted plug-in, the SAE monitors the states of router interfaces associated with backbone congestion points. The SAE sends relevant data to SRC-ACP through the SRC-ACP's remote interface.

7. When the subscriber requests activation of a service subscription (either through the SAE core API or automatically for activate-on-login services), the SAE notifies SRC-ACP to authorize and track the service usage.

   a. The SAE sends the requested bandwidth to SRC-ACP.

   b. SRC-ACP authorizes or denies service activation.

      If SRC-ACP authorizes the service activation, the SAE activates the service and sends a tracking event to SRC-ACP. SRC-ACP updates the current bandwidth for all congestion points with the requested bandwidth.

If SRC-ACP authorizes the service activation with state synchronization enabled, SRC-ACP reserves the requested bandwidth on all congestion points until the reservation expires. You can specify the reservation timeout value when configuring SRC-ACP operation.

- For each congestion point, SRC-ACP verifies whether:

  (current bw + reserved bw + requested bw) >
  [provisioned bw - (background bw x tuning factor)]

  If the desired bandwidth exceeds the allocated bandwidth, SRC-ACP denies service activation.

- When SRC-ACP receives a service start tracking event, the requested bandwidth is committed. That is, for each congestion point, the requested bandwidth reservation is removed and the requested bandwidth is added to the current bandwidth.

- When the bandwidth reservation expires, the reserved bandwidth is released.

If SRC-ACP does not authorize the service activation, the SAE delivers a message detailing the reason to the originator of the activation request.

SRC-ACP distinguishes between bandwidth exceeded on the subscriber interface (first congestion point) and bandwidth exceeded on a network interface by sending two different messages back to the SAE. In the first case, the subscriber may resolve the bandwidth problem by deactivating another service.

8. When a service is deactivated (either through the SAE core API or because a session times out), SRC-ACP updates the current bandwidth for all congestion points by removing the original requested bandwidth.

9. SRC-ACP stores all information about subscribers, services, and congestion points in a set of files.

   SRC-ACP continually adds data to these files, but does not delete old data. Consequently, the sizes of the files continue to increase. SRC-ACP does, however, reorganize the files when the sum of their sizes increments by a specified value. Reorganizing the files reduces their sizes. You can also reorganize the files by using the SRC CLI (see *Reorganizing the File That Contains ACP Data* on page 325.)

Redundancy

You can configure SRC-ACP redundancy for a region of the network by installing SRC-ACP on two different hosts, installing a naming service application on the SAE, and connecting both SRC-ACP hosts to the SAE (see Figure 45 on page 274). One SRC-ACP acts as the primary application, and the other as the secondary application.

NOTE: Both SRC-ACPs in a redundant pair must operate in the same mode. You cannot configure an SRC-ACP in edge mode and an SRC-ACP in backbone mode as a redundant pair.

The primary and secondary SRC-ACPs communicate with each other through a CORBA interface. When you start each SRC-ACP (see *Starting SRC-ACP* on page 325), it will register its redundancy CORBA interface with the naming service application, and import the interface for the other SRC-ACP from the naming service application.

Each SRC-ACP continuously monitors the other's availability. The primary SRC-ACP receives data from the SAE and sends any changes to the secondary SRC-ACP. If the secondary SRC-ACP is unavailable, the primary SRC-ACP caches the data to send when the secondary SRC-ACP becomes available.

If the primary SRC-ACP becomes unavailable, the secondary SRC-ACP immediately notifies the naming service application and assumes the primary role. If the former primary SRC-ACP recovers very quickly, it will again assume the primary role. However, if the former primary SRC-ACP recovers more slowly, it will assume the primary role only if the former secondary SRC-ACP becomes unavailable.

Fault Recovery

If the SAE cannot reach SRC-ACP, the SAE will deny all service activation requests. As soon as it reaches SRC-ACP, the SAE again sends authorization requests to SRC-ACP.

SRC-ACP keeps the state of the congestion points in persistent storage, and if SRC-ACP becomes unavailable, the service authorization can continue in the correct state. Because service activation requests are automatically denied when the SAE cannot reach SRC-ACP, SRC-ACP does not miss any active service sessions. The SAE will resend all service deactivation requests after SRC-ACP is reachable again.

SRC-ACP monitors SAE synchronization events for information about VR availability and SAE availability. If a VR reboots or an SAE becomes unavailable, SRC-ACP updates the states of congestion points associated with those devices accordingly.

If the SAE becomes unavailable, the router will automatically reestablish connection to either the redundant SAE or, if a redundant SAE is not available, to the original SAE when it again becomes available. The new SAE notifies SRC-ACP that the original SAE failed and specifies which subscriber and service sessions were logged during this time. SRC-ACP uses this information to update its state.

State Synchronization

You can configure SRC-ACP to synchronize states with the SAE.

If state synchronization is enabled, the current state can be transferred when SRC-ACP has started up or lost its state. SRC-ACP does not have to keep a local and persistent copy of the state. However, SRC-ACP requires additional bandwidth to transfer state information that can affect performance.

Both SRC-ACP redundancy and state synchronization can be enabled at the same time. In this situation, the primary and secondary SRC-ACPs are set up as a community and will communicate with each other to determine the primary SRC-ACP. The primary SRC-ACP registers its interoperable object reference (IOR) with the SAE so that the SAE will communicate only with the primary SRC-ACP. When the primary SRC-ACP becomes unavailable, the secondary SRC-ACP assumes the role of the primary SRC-ACP and performs state synchronization if necessary.

API for ACP

You can develop your own application to update information about subscribers and congestion points for SRC-ACP. The application can call one method to interact with SRC-ACP. This method is called:

update (in RemoteUpdateType rut, in TagValueList attrs)

The method takes a property-value pair and passes the information to SRC-ACP. For information about the properties and values you can pass to SRC-ACP, see the file *acpPlugin.idl* in the folder *SDK/idl* in the SRC software distribution.

To create an application that updates SRC-ACP remotely:

1. Compile the IDL file, and generate the code in the language in which you want to write the application.
2. Write the application, and include the generated code for the IDL file.
3. Use the CORBA object reference defined in the property ACP.syncRateAdaptor.ior to send data from the application to SRC-ACP.

For information about the interfaces, properties, and methods available in the CORBA remote API for ACP, see the documentation in the SRC software distribution at *SDK/doc/idl/acp/html/index.html*.

Chapter 21

Configuring Admission Control with the SRC CLI

This chapter describes how to use the SRC command-line interface (CLI) to configure the SRC Admission Control Plug-In (SRC-ACP) application for use in the SRC network. You can use the CLI to configure SRC-ACP on a Solaris platform or on a C-series Controller.

You can also use SRC configuration applications to configure SRC-ACP on a Solaris platform. See *Chapter 26, Providing Admission Control with SRC-ACP on a Solaris Platform*.

Topics in this chapter include:

Configuration Statements for SRC-ACP

Use the following configuration statements to configure SRC-ACP at the [edit] hierarchy level:

```
shared acp configuration acp-options {
    backup-directory backup-directory;
    mode (edge | backbone | dual);
    event-cache-size event-cache-size;
    overload-method overload-method;
    reservation-timeout reservation-timeout;
    congestion-point-auto-completion;
    tuning-factor tuning-factor;
    subscriber-bandwidth-exceed-message subscriber-bandwidth-exceed-message;
    network-bandwidth-exceed-message network-bandwidth-exceed-message;
    backup-database-maximum-size backup-database-maximum-size;
    remote-update-database-index-keys remote-update-database-index-keys;
    interface-tracking-filter interface-tracking-filter;
    state-sync-bulk-size state-sync-bulk-size;
}

shared acp configuration corba {
   acp-ior acp-ior;
   remote-update-ior remote-update-ior;
}

shared acp configuration ldap service-data {
    edge-congestion-point-dn edge-congestion-point-dn;
    backbone-congestion-point-dn backbone-congestion-point-dn;
    reload-congestion-points;
    congestion-points-eventing;
    server-address server-address;
    server-port server-port;
    dn dn;
    principal principal;
    password password;
    event-dn event-dn;
    directory-eventing;
    polling-interval polling-interval;
}

shared acp configuration ldap subscriber-data {
    congestion-points-eventing;
    server-address server-address;
    server-port server-port;
    dn dn;
    principal principal;
    password password;
    event-dn event-dn;
    directory-eventing;
    polling-interval polling-interval;
}

shared acp configuration logger name ...

shared acp configuration logger name file {
    filter filter;
    filename filename;
    rollover-filename rollover-filename;
    maximum-file-size maximum-file-size;
}

shared acp configuration logger name syslog {
    filter filter;
    host host;
    facility facility;
    format format;
}

shared acp configuration redundancy {
    enable-redundancy;
    local-ior local-ior;
    remote-ior remote-ior;
    ignore-user-tracking-out-of-sync;
    community-heartbeat community-heartbeat;
    community-acquire-timeout community-acquire-timeout;
    community-blackout-timeout community-blackout-timeout;
    redundant-naming-service redundant-naming-service;
}

shared acp configuration scripts-and-classification {
    script-factory-class script-factory-class;
    classification-factory-class classification-factory-class;
    classification-script classification-script;
    congestion-point-profile-script congestion-point-profile-script;
    extension-path extension-path;
}

shared admission-control device name {
    description description;
}

shared admission-control device name interface name {
    description description;
    upstream-provisioned-rate upstream-provisioned-rate;
    downstream-provisioned-rate downstream-provisioned-rate;
    upstream-background-bandwidth upstream-background-bandwidth;
    downstream-background-bandwidth downstream-background-bandwidth;
    action-type (url | python | java-class | java-archive);
    action-class-name action-class-name;
    action-file-url action-file-url;
    action-parameters [action-parameters...];
}

shared congestion-points profile name {
    interface [interface...];
}

slot number acp {
    java-runtime-environment java-runtime-environment;
    java-heap-size java-heap-size;
    java-garbage-collection-options java-garbage-collection-options;
    base-dn base-dn;
    snmp-agent;
    shared shared;
}

slot number acp initial {
    static-dn static-dn;
    dynamic-dn dynamic-dn;
}

slot number acp initial directory-connection {
    url url;
    backup-urls [backup-urls...];
    principal principal;
    credentials credentials;
    protocol (ldaps);
    timeout timeout;
    check-interval check-interval;
    blacklist;
    snmp-agent;
}

slot number acp initial directory-eventing {
    eventing;
    signature-dn signature-dn;
    polling-interval polling-interval;
    event-base-dn event-base-dn;
    dispatcher-pool-size dispatcher-pool-size;
}
```

For detailed information about each configuration statement, see the *SRC-PE CLI Command Reference.*

Configuring SRC-ACP

To use SRC-ACP in the SRC network, you must perform some configuration. For information about these configuration procedures, see:

1. (Optional) Creating Grouped Configurations for SRC-ACP on page 287
2. Configuring Local Properties for SRC-ACP on page 288
3. Configuring the SAE for SRC-ACP on page 291
4. Configuring SRC-ACP Properties on page 294
5. (Edge and dual mode only) Configuring SRC-ACP to Manage the Edge Network on page 306
6. (Backbone and dual mode only) *Configuring SRC-ACP to Manage the Backbone Network* on page 310
7. *Starting SRC-ACP* on page 325

You can automate and scale the configuration of congestion points using congestion point classification. For more information, see *Chapter 22, Configuring Congestion Point Classification with the SRC CLI*.

Creating Grouped Configurations for SRC-ACP

We recommend that you configure SRC-ACP within a group. When you create a configuration group, the software creates a configuration with default values filled in.

Configuration groups allow you to share the SRC-ACP configuration with different SRC-ACP instances in the SRC network. You can also set up different configurations for different instances.

You can then create a grouped SRC-ACP configuration that is shared with some SRC-ACP instances. For example, if you create two different SRC-ACP groups called config1 and config2 within the shared SRC-ACP configuration, you could select the SRC-ACP configuration that should be associated with a particular SRC-ACP instance.

Configuring an SRC-ACP Group

Use the shared option of the slot *number* acp statement to select the group for an SRC-ACP instance as part of the local configuration. Use the shared acp group *name* statements to configure the group.

To select and configure a group:

1. From configuration mode, select a group for an SRC-ACP instance. For example, to select a group called config1 in the path /:

   ```
   [edit]
   user@host# set slot 0 acp shared /config1
   ```

   For more information, see *Configuring Basic Local Properties for SRC-ACP* on page 288.

2. Commit the configuration.

   ```
   [edit]
   user@host# commit
   commit complete.
   ```

3. From configuration mode, configure a group. For example, to configure a group called config1, specify the group as part of the SRC-ACP configuration.

   ```
   [edit]
   user@host# edit shared acp group config1 ?
   Possible completions:
     <[Enter]>            Execute this command
   > configuration
   > congestion-point-classifier
   > group                Group of ACP configuration properties
     |                    Pipe through a command
   ```

   For more information, see *Configuring SRC-ACP Properties* on page 294.

Configuring Local Properties for SRC-ACP

To configure the local properties for SRC-ACP:

1. Configure basic local properties, including Java heap memory.

   See *Configuring Basic Local Properties for SRC-ACP* on page 288.

2. Configure initial properties, including directory connection and directory eventing properties.

   See *Configuring Initial Properties for SRC-ACP* on page 289.

   See *Configuring Directory Connection Properties for SRC-ACP* on page 290.

   See *Configuring Initial Directory Eventing Properties for SRC-ACP* on page 291.

Configuring Basic Local Properties for SRC-ACP

Use the following configuration statements to configure basic local properties for SRC-ACP:

```
slot number acp {
    java-runtime-environment java-runtime-environment;
    java-heap-size java-heap-size;
    java-garbage-collection-options java-garbage-collection-options;
    base-dn base-dn;
    snmp-agent;
    shared shared;
}
```

To configure basic local properties:

1. From configuration mode, access the configuration statement that configures the local properties.

   ```
   user@host# edit slot 0 acp
   ```

2. Specify the basic local properties for ACP.

   ```
   [edit slot 0 acp]
   user@host# set ?
   ```

For more information about configuring local properties for the SRC components, see *SRC-PE Getting Started Guide, Chapter 30, Configuring Local Properties with the SRC CLI*.

3. Configure the garbage collection functionality of the Java Virtual Machine.

```
[edit slot 0 acp]
user@host# set java-garbage-collection-options java-garbage-collection-options
```

4. Select an SRC-ACP group configuration.

```
[edit slot 0 acp]
user@host# set shared shared
```

For more information, see *Creating Grouped Configurations for SRC-ACP* on page 287.

5. (Optional) Verify your configuration.

```
[edit slot 0 acp]
user@host# show
shared /config;
initial {
  directory-connection {
    url ldap://127.0.0.1:389/;
    principal cn=conf,o=Operators,<base>;
    credentials ********;
  }
  directory-eventing {
    eventing;
    polling-interval 30;
  }
}
```

Configuring Initial Properties for SRC-ACP

Use the following configuration statements to configure initial properties for SRC-ACP:

```
slot number acp initial {
    static-dn static-dn;
    dynamic-dn dynamic-dn;
}
```

To configure initial local properties:

1. From configuration mode, access the configuration statement that configures the initial properties.

```
user@host# edit slot 0 acp initial
```

2. Specify the properties for SRC-ACP.

```
[edit slot 0 acp initial]
user@host# set ?
```

For more information about configuring local properties for the SRC components, see *SRC-PE Getting Started Guide, Chapter 30, Configuring Local Properties with the SRC CLI*.

3. (Optional) Verify your configuration.

```
[edit slot 0 acp initial]
user@host# show
```

Configuring Directory Connection Properties for SRC-ACP

Use the following configuration statements to configure directory connection properties for SRC-ACP:

```
slot number acp initial directory-connection {
    url url;
    backup-urls [backup-urls...];
    principal principal;
    credentials credentials;
    protocol (ldaps);
    timeout timeout;
    check-interval check-interval;
    blacklist;
    snmp-agent;
}
```

To configure directory connection properties:

1. From configuration mode, access the configuration statement that configures the directory connection properties.

```
user@host# edit slot 0 acp initial directory-connection
```

2. Specify the properties for ACP.

```
[edit slot 0 acp initial directory-connection]
user@host# set ?
```

For more information about configuring local properties for the SRC components, see *SRC-PE Getting Started Guide, Chapter 30, Configuring Local Properties with the SRC CLI*.

3. (Optional) Verify your configuration.

```
[edit slot 0 acp initial directory-connection]
user@host# show
url ldap://127.0.0.1:389/;
principal cn=conf,o=Operators,<base>;
credentials ********;
```

Configuring Initial Directory Eventing Properties for SRC-ACP

Use the following configuration statements to configure directory eventing properties for SRC-ACP:

```
slot number acp initial directory-eventing {
    eventing;
    signature-dn signature-dn;
    polling-interval polling-interval;
    event-base-dn event-base-dn;
    dispatcher-pool-size dispatcher-pool-size;
}
```

To configure initial directory eventing properties:

1. From configuration mode, access the configuration statement that configures the local properties.

   ```
   user@host# edit slot 0 acp initial eventing
   ```

2. Specify the initial directory eventing properties for SRC-ACP.

   ```
   [edit slot 0 acp initial directory-eventing]
   user@host# set ?
   ```

   For more information about configuring local properties for the SRC components, see *SRC-PE Getting Started Guide, Chapter 30, Configuring Local Properties with the SRC CLI*.

3. (Optional) Verify your configuration.

   ```
   [edit slot 0 acp initial directory-eventing]
   user@host# show
   eventing;
   polling-interval 30;
   ```

Configuring the SAE for SRC-ACP

You must configure the SAE to recognize SRC-ACP by adding information about SRC-ACP to the SAE properties. To do so:

1. Configure SRC-ACP as an external plug-in for the SAE.
2. Configure event publishers.
3. (Backbone and dual mode only) Optionally, configure a hosted plug-in that monitors the state of interfaces on VRs.

Configuring SRC-ACP as an External Plug-In

To configure an external plug-in for the SAE:

1. From configuration mode, access the configuration statement that configures the external plug-ins.

   ```
   user@host# edit shared sae configuration plug-ins name name external
   ```

2. Specify the the plug-in attributes.

   ```
   [edit shared sae configuration plug-ins name name external]
   user@host# set attr ?
   ```

   For edge and dual modes—upstream-bandwidth, downstream-bandwidth, service-name, router-name, login-name, user-dn, port-id, session-id, user-ip-address, nas-ip, user-session-id, event-time

   For backbone mode—upstream-bandwidth, downstream-bandwidth, service-name, router-name, session-id, nas-ip, event-time

   For more information about configuring plug-in attributes, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 9, Configuring Internal, External, and Synchronization Plug-Ins with the SRC CLI*.

Configuring Event Publishers

You must configure the SAE to publish the following types of events to SRC-ACP:

- (Edge and dual mode only) Global subscriber tracking
- Global service authorization
- Global service tracking

For information about configuring event publishers, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 11, Configuring Accounting and Authentication Plug-Ins with the SRC CLI*. Identify the instance of SRC-ACP by the name of the host on which you configured it.

Configuring the SAE to Monitor Interfaces for Congestion Points

NOTE: Configure this feature only if SRC-ACP is in backbone or dual mode.

The SAE uses a hosted internal plug-in to monitor the state of interfaces on a VR for backbone congestion points. If a subscriber tries to activate a service on an interface that is unavailable, the SAE denies the request. The plug-in also monitors the directory for new backbone congestion points.

When this plug-in initializes, it reads all the backbone services from the directory and generates a list of the DNs (network interfaces) of the backbone congestion points. The SAE sends interface tracking events, which contain the names of the interfaces, VRs, and routers to this plug-in. For this feature to work correctly, the interface, VR, and router must be configured (see *Configuring Network Interfaces in the Directory for the Backbone Network* on page 310).

To configure the ACP interface listener as an internal plug-in for the SAE:

1. From configuration mode, access the configuration statement that configures the ACP interface listener.

   ```
   user@host# edit shared sae configuration plug-ins name name
   acp-interface-listener
   ```

2. Specify the IP address or name of the host that supports the directory that contains backbone service definitions and network interfaces.

   ```
   [edit shared sae configuration plug-ins name name acp-interface-listener]
   user@host# set ldap-server ldap-server
   ```

3. Specify the DN of the directory entry that defines the username with which the plug-in accesses the directory.

   ```
   [edit shared sae configuration plug-ins name name acp-interface-listener]
   user@host# set bind-dn bind-dn
   ```

4. Specify the password with which the plug-in accesses the directory.

   ```
   [edit shared sae configuration plug-ins name name acp-interface-listener]
   user@host# set bind-password bind-password
   ```

5. Specify whether the connection to the directory uses secure LDAP. If you do not configure a security protocol, plain socket is used.

   ```
   [edit shared sae configuration plug-ins name name acp-interface-listener]
   user@host# set ldaps
   ```

6. Specify the DN at which SRC-ACP stores backbone congestion points.

   ```
   [edit shared sae configuration plug-ins name name acp-interface-listener]
   user@host# set congestion-points-base-dn congestion-points-base-dn
   ```

7. Specify the DN at which SRC-ACP stores edge congestion points.

   ```
   [edit shared sae configuration plug-ins name name acp-interface-listener]
   user@host# set admission-control-base-dn admission-control-base-dn
   ```

8. (Optional) Specify the maximum time that the plug-in waits for the router to respond.

   ```
   [edit shared sae configuration plug-ins name name acp-interface-listener]
   user@host# set timeout timeout
   ```

9. Specify the object reference for the ACP plug-in, as defined by the object reference for SRC-ACP (see the acp-ior option in *Configuring CORBA Interfaces* on page 299).

```
[edit shared sae configuration plug-ins name name acp-interface-listener]
user@host# set acp-remote-corba-ior acp-remote-corba-ior
```

10. (Optional) Verify your configuration.

```
[edit shared sae configuration plug-ins name name acp-interface-listener]
user@host# show
```

Configuring SRC-ACP Properties

To configure SRC-ACP properties, perform these tasks:

- Configuring Logging Destinations for SRC-ACP on page 294
- Configuring SRC-ACP Operation on page 296
- *Configuring CORBA Interfaces* on page 299
- Configuring SRC-ACP Redundancy on page 300
- Configuring Connections to the Subscribers' Directory on page 302
- Configuring Connections to the Services' Directory on page 303
- Configuring SRC-ACP Scripts and Classification on page 305

Configuring Logging Destinations for SRC-ACP

Use the following configuration statements to configure logging destinations for SRC-ACP:

```
shared acp configuration logger name ...

shared acp configuration logger name file {
    filter filter;
    filename filename;
    rollover-filename rollover-filename;
    maximum-file-size maximum-file-size;
}

shared acp configuration 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 file-1 is configured in the config group.

   ```
   user@host# edit shared acp group config configuration logger file-1 file
   ```

2. Specify the properties for the logging destination.

   ```
   [edit shared acp group config configuration logger file-1 file]
   user@host# set ?
   ```

   For more information about configuring properties for the logging destination, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 3, Configuring Logging for SRC Components with the CLI*.

3. (Optional) Verify your configuration.

   ```
   [edit shared acp group config configuration logger file-1 file]
   user@host# show
   filename var/log/acp_debug.log;
   rollover-filename var/log/acp_debug.alt;
   ```

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 syslog-1 is configured in the config group.

   ```
   user@host# edit shared acp group config configuration logger syslog-1 syslog
   ```

2. Specify the properties for the logging destination.

   ```
   [edit shared acp group config configuration logger syslog-1 syslog]
   user@host# set ?
   ```

   For more information about configuring properties for the logging destination, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 3, Configuring Logging for SRC Components with the CLI*.

3. (Optional) Verify your configuration.

   ```
   [edit shared acp group config configuration logger syslog-1 syslog]
   user@host# show
   filter /error-;
   host loghost;
   ```

Configuring SRC-ACP Operation

Use the following configuration statements to configure how SRC-ACP operates:

```
shared acp configuration acp-options {
    backup-directory backup-directory;
    mode (edge | backbone | dual);
    event-cache-size event-cache-size;
    overload-method overload-method;
    reservation-timeout reservation-timeout;
    congestion-point-auto-completion;
    tuning-factor tuning-factor;
    subscriber-bandwidth-exceed-message subscriber-bandwidth-exceed-message;
    network-bandwidth-exceed-message network-bandwidth-exceed-message;
    backup-database-maximum-size backup-database-maximum-size;
    remote-update-database-index-keys remote-update-database-index-keys;
    interface-tracking-filter interface-tracking-filter;
    state-sync-bulk-size state-sync-bulk-size;
}
```

To configure SRC-ACP operation:

1. From configuration mode, access the configuration statement that configures SRC-ACP operation. In this sample procedure, the SRC-ACP operating properties are configured in the config group.

   ```
   user@host# edit shared acp group config configuration acp-options
   ```

2. Specify the folder that stores backup information about subscribers, services, and congestion points.

   ```
   [edit shared acp group config configuration acp-options]
   user@host# set backup-directory
   ```

3. Specify the regions of the network that SRC-ACP manages.

   ```
   [edit shared acp group config configuration acp-options]
   user@host# set mode (edge | backbone | dual)
   ```

4. Specify the number of plug-in events from the SAE that SRC-ACP can store in its cache.

   ```
   [edit shared acp group config configuration acp-options]
   user@host# set event-cache-size event-cache-size
   ```

5. Specify how SRC-ACP deals with situations in which the components exceed the allocated bandwidth because the service was activated after the authorization was granted.

   ```
   [edit shared acp group config configuration acp-options]
   user@host# set overload-method overload-method
   ```

   If you specify -1, SRC-ACP ignores overload. An integer greater than or equal to 0 specifies the bandwidth (in bits per second) by which the maximum may be exceeded.

6. Specify the time to wait before a bandwidth reservation expires. The reserved bandwidth is reclaimed by SRC-ACP when the reservation expires.

   ```
   [edit shared acp group config configuration acp-options]
   user@host# set reservation-timeout reservation-timeout
   ```

7. Specify whether SRC-ACP uses the information acquired from the router to determine the congestion points.

   ```
   [edit shared acp group config configuration acp-options]
   user@host# set congestion-point-auto-completion
   ```

8. Specify the factors that compensate for actual use of bandwidth, as opposed to allocated bandwidth.

   ```
   [edit shared acp group config configuration acp-options]
   user@host# set tuning-factor tuning-factor
   ```

9. Specify the error message that SRC-ACP sends when the subscriber exceeds the allocated bandwidth.

   ```
   [edit shared acp group config configuration acp-options]
   user@host# set subscriber-bandwidth-exceed-message
   subscriber-bandwidth-exceed-message
   ```

10. Specify the error message that SRC-ACP sends when traffic flow exceeds the allocated bandwidth on an interface between the subscriber and the router.

    ```
    [edit shared acp group config configuration acp-options]
    user@host# set network-bandwidth-exceed-message
    network-bandwidth-exceed-message
    ```

11. Specify the value by which the sum of the sizes of the files that contain SRC-ACP data can increment before SRC-ACP reorganizes the files.

    ```
    [edit shared acp group config configuration acp-options]
    user@host# set backup-database-maximum-size backup-database-maximum-size
    ```

    Choose a value that is significantly lower than the capacity of the machine's hard disk.

12. Specify the values to look for in the configuration data. Specifying index keys can improve performance by filtering the data.

    ```
    [edit shared acp group config configuration acp-options]
    user@host# set remote-update-database-index-keys
    remote-update-database-index-keys
    ```

    The value is a list of attributes, separated by commas. An attribute is one of the following text strings:

    - accountingId—Value of directory attribute accountingUserId.
    - dhcpPacket—Content of the DHCP discover request.
    - hostname— Name of the host on which the SAE is installed.

- ifIndex—SNMP index of the interface. This attribute is not supported on JUNOS routing platforms.
- ifRadiusClass—RADIUS class attribute on the JUNOSe interface. This attribute is not supported on JUNOS routing platforms.
- ifSessionId—Identifier for RADIUS accounting on the JUNOSe interface. This attribute is not supported on JUNOS routing platforms.
- interfaceAlias—Alias of the interface; that is, the IP description in the interface configuration.
- interfaceDescr—SNMP description of the interface.
- interfaceName—Name of the interface.
- loginName—Subscriber's login name.
- nasInetAddress—IP address of the router; using a byte array instead of an integer.
- nasPort—NAS port used by the router to identify the interface to RADIUS.
- portId—Identifier of VLAN or virtual circuit. For a virtual circuit, use the format <VPI>/<VCI>. This attribute is not supported on JUNOS routing platforms.
    - <VPI>—Virtual path identifier
    - <VCI>—Virtual connection identifier
- primaryUserName—PPP login name or the public DHCP username. This attribute is not supported on JUNOS routing platforms.
- routerName—Name of the virtual router in the format <virtualRouter>@<router>.
    - <virtualRouter>—Virtual router name
    - <router>—Router name
- routerType—Type of router driver.
- userInetAddress—IP address of the subscriber that uses a byte array instead of an integer.
- userMacAddress—MAC address of the DHCP subscriber. This attribute is not supported on JUNOS routing platforms.
- userRadiusClass—RADIUS class attribute of the subscriber session for a service. This attribute can occur multiple times and can be returned by an authorization plug-in.
- userType—Type of subscriber.

13. Specify the interface tracking event to be ignored by SRC-ACP.

```
[edit shared acp group config configuration acp-options]
user@host# set interface-tracking-filter interface-tracking-filter
```

The value is filter strings in the format of a list of <attribute> = <value> pairs. The filter strings can be contained within query operations.

- <attribute>—Name of an attribute for an interface tracking event. See value for the **remote-update-database-index-keys** option described in step 12.
- <value>—Filtering string of the following types:
    - *—Any value
    - Explicit string—Any value matching the specified string (not case-sensitive)
    - String containing an asterisk—Any value containing the specified string (not case-sensitive)
- To perform query operations on filter strings, you can use the following values in your filter strings:
    - ()—Match no objects.
    - (*)—Match all objects.
    - (& <filter> <filter> ...)—Performs logical AND operation on filter strings; true if all filter strings match.
    - (| <filter> <filter> ...)—Performs logical OR operation on filter strings; true if at least one filter string matches.
    - (! <filter>)—Performs logical NOT operation on filter string; true if the filter string does not match.

14. (Optional) Specify the number of events the SAE sends to SRC-ACP in a single method call during state synchronization.

```
[edit shared acp group config configuration acp-options]
user@host# set state-sync-bulk-size state-sync-bulk-size
```

15. (Optional) Verify your configuration.

```
[edit shared acp group config configuration acp-options]
user@host# show
```

Configuring CORBA Interfaces

Use the following configuration statements to configure CORBA interfaces for SRC-ACP:

```
shared acp configuration corba {
    acp-ior acp-ior;
    remote-update-ior remote-update-ior;
}
```

To configure CORBA interfaces:

1. From configuration mode, access the configuration statement that configures CORBA interfaces for SRC-ACP. In this sample procedure, the CORBA interfaces are configured in the config group.

   ```
   user@host# edit shared acp group config configuration corba
   ```

2. Export the object reference for SRC-ACP through either a local file or a Common Object Services (COS) naming service.

   ```
   [edit shared acp group config configuration corba]
   user@host# set acp-ior acp-ior
   ```

3. Specify the object reference for the ACP external interface.

   ```
   [edit shared acp group config configuration corba]
   user@host# set remote-update-ior remote-update-ior
   ```

4. (Optional) Verify your configuration.

   ```
   [edit shared acp group config configuration corba]
   user@host# show
   acp-ior file:///var/acp/acp.ior;
   remote-update-ior file:///var/acp/sra.ior;
   ```

Configuring SRC-ACP Redundancy

Use the following configuration statements to configure SRC-ACP redundancy and state synchronization with the SAE:

```
shared acp configuration redundancy {
    enable-redundancy;
    local-ior local-ior;
    remote-ior remote-ior;
    ignore-user-tracking-out-of-sync;
    community-heartbeat community-heartbeat;
    community-acquire-timeout community-acquire-timeout;
    community-blackout-timeout community-blackout-timeout;
    redundant-naming-service redundant-naming-service;
}
```

To configure SRC-ACP redundancy and state synchronization with the SAE:

1. From configuration mode, access the configuration statement that configures SRC-ACP redundancy. In this sample procedure, the properties are configured in the config group.

   ```
   user@host# edit shared acp group config configuration redundancy
   ```

2. (Optional) Enable SRC-ACP redundancy.

   ```
   [edit shared acp group config configuration redundancy]
   user@host# set enable-redundancy
   ```

3. Export the object reference for this SRC-ACP (local interface) through a Common Object Services (COS) naming service in a redundant SRC-ACP configuration.

   ```
   [edit shared acp group config configuration redundancy]
   user@host# set local-ior local-ior
   ```

4. Resolves the object reference for the other SRC-ACP (remote interface) through a Common Object Services (COS) naming service in a redundant SRC-ACP configuration. For redundancy, the remote IOR value of one SRC-ACP must match the local IOR value of the other SRC-ACP.

   ```
   [edit shared acp group config configuration redundancy]
   user@host# set remote-ior remote-ior
   ```

5. (Optional) Specify whether user tracking events should be ignored when they raise an OutOfSync exception to the SAE when state synchronization is enabled. SRC-ACP raises an OutOfSync exception when SRC-ACP handles service tracking or authentication events without receiving a user start event first.

   ```
   [edit shared acp group config configuration redundancy]
   user@host# set ignore-user-tracking-out-of-sync
   ```

6. (Optional) Specify the time interval for community members to check each other’s availability when both redundancy and state synchronization are enabled.

   ```
   [edit shared acp group config configuration redundancy]
   user@host# set community-heartbeat community-heartbeat
   ```

7. (Optional) Specify the time to wait before trying to reacquire the distributed lock when both redundancy and state synchronization are enabled.

   ```
   [edit shared acp group config configuration redundancy]
   user@host# set community-acquire-timeout community-acquire-timeout
   ```

8. (Optional) Specify the time to wait before regaining control when both redundancy and state synchronization are enabled.

   ```
   [edit shared acp group config configuration redundancy]
   user@host# set community-blackout-timeout community-blackout-timeout
   ```

9. Export the object reference for the backup naming service through a local file or COS naming service in a redundant SRC-ACP configuration. The primary SRC-ACP registers the IOR and redundancy IOR to both naming services, while the secondary SRC-ACP registers the redundancy IOR to both naming services.

   ```
   [edit shared acp group config configuration redundancy]
   user@host# set redundant-naming-service redundant-naming-service
   ```

10. (Optional) Verify your configuration.

    ```
    [edit shared acp group config configuration redundancy]
    user@host# show
    ```

Configuring Connections to the Subscribers' Directory

Use the following configuration statements to configure how SRC-ACP connects to the directory that contains subscriber information:

```
shared acp configuration ldap subscriber-data {
    congestion-points-eventing;
    server-address server-address;
    server-port server-port;
    dn dn;
    principal principal;
    password password;
    event-dn event-dn;
    directory-eventing;
    polling-interval polling-interval;
}
```

To configure connections to the directory that stores subscriber information:

1. From configuration mode, access the configuration statement that configures SRC-ACP connections to the subscribers' directory. In this sample procedure, the connections are configured in the config group.

   ```
   user@host# edit shared acp group config configuration ldap subscriber-data
   ```

2. (Optional) Enable directory eventing for congestion points.

   ```
   [edit shared acp group config configuration ldap subscriber-data]
   user@host# set congestion-points-eventing
   ```

3. Specify the list of primary and redundant servers that manage data for subscribers.

   ```
   [edit shared acp group config configuration ldap subscriber-data]
   user@host# set server-address server-address
   ```

4. Specify the TCP port for the directory.

   ```
   [edit shared acp group config configuration ldap subscriber-data]
   user@host# set server-port server-port
   ```

5. Specify the DN of the root of the directory.

   ```
   [edit shared acp group config configuration ldap subscriber-data]
   user@host# set dn dn
   ```

6. Specify the DN used to authorize connections to the directory.

   ```
   [edit shared acp group config configuration ldap subscriber-data]
   user@host# set principal principal
   ```

7. Specify the password used to authorize connections to the directory.

   ```
   [edit shared acp group config configuration ldap subscriber-data]
   user@host# set password password
   ```

8. Specify the DN of the directory that contains event information.

```
[edit shared acp group config configuration ldap subscriber-data]
user@host# set event-dn event-dn
```

9. (Optional) Enable directory eventing.

```
[edit shared acp group config configuration ldap subscriber-data]
user@host# set directory-eventing
```

10. Specify the time interval at which the SRC component polls the directory.

```
[edit shared acp group config configuration ldap subscriber-data]
user@host# set polling-interval polling-interval
```

11. (Optional) Verify your configuration.

```
[edit shared acp group config configuration ldap subscriber-data]
user@host# show
```

Configuring Connections to the Services' Directory

Use the following configuration statements to configure how SRC-ACP connects to the directory that contains information about services:

```
shared acp configuration ldap service-data {
    edge-congestion-point-dn edge-congestion-point-dn;
    backbone-congestion-point-dn backbone-congestion-point-dn;
    reload-congestion-points;
    congestion-points-eventing;
    server-address server-address;
    server-port server-port;
    dn dn;
    principal principal;
    password password;
    event-dn event-dn;
    directory-eventing;
    polling-interval polling-interval;
}
```

To configure connections to the directory that stores service information:

1. From configuration mode, access the configuration statement that configures SRC-ACP connections to the services' directory. In this sample procedure, the connections are configured in the config group.

```
user@host# edit shared acp group config configuration ldap service-data
```

2. Specify the DN of the directory that contains information about network interfaces for edge congestion points.

```
[edit shared acp group config configuration ldap service-data]
user@host# set edge-congestion-point-dn edge-congestion-point-dn
```

3. Specify the DN of the directory that contains information about network interfaces for backbone congestion point objects.

   ```
   [edit shared acp group config configuration ldap service-data]
   user@host# set backbone-congestion-point-dn backbone-congestion-point-dn
   ```

4. (Optional) Specify whether SRC-ACP detects changes in the backbone congestion point for a service while SRC-ACP is operative.

   ```
   [edit shared acp group config configuration ldap service-data]
   user@host# set reload-congestion-points
   ```

   Set this value only when you want to modify a congestion point.

5. (Optional) Enable directory eventing for congestion points.

   ```
   [edit shared acp group config configuration ldap service-data]
   user@host# set congestion-points-eventing
   ```

6. Specify the list of primary and redundant servers that manage data for subscribers.

   ```
   [edit shared acp group config configuration ldap service-data]
   user@host# set server-address server-address
   ```

7. Specify the TCP port for the directory.

   ```
   [edit shared acp group config configuration ldap service-data]
   user@host# set server-port server-port
   ```

8. Specify the DN of the root of the directory.

   ```
   [edit shared acp group config configuration ldap service-data]
   user@host# set dn dn
   ```

9. Specify the DN used to authorize connections to the directory.

   ```
   [edit shared acp group config configuration ldap service-data]
   user@host# set principal principal
   ```

10. Specify the password used to authorize connections to the directory.

    ```
    [edit shared acp group config configuration ldap service-data]
    user@host# set password password
    ```

11. Specify the DN of the directory that contains event information.

    ```
    [edit shared acp group config configuration ldap service-data]
    user@host# set event-dn event-dn
    ```

12. (Optional) Enable directory eventing.

    ```
    [edit shared acp group config configuration ldap service-data]
    user@host# set directory-eventing
    ```

13. Specify the time interval at which the SRC component polls the directory.

```
[edit shared acp group config configuration ldap service-data]
user@host# set polling-interval polling-interval
```

14. (Optional) Verify your configuration.

```
[edit shared acp group config configuration ldap service-data]
user@host# show
```

Configuring SRC-ACP Scripts and Classification

Use the following configuration statements to configure SRC-ACP scripts and classification:

```
shared acp configuration scripts-and-classification {
    script-factory-class script-factory-class;
    classification-factory-class classification-factory-class;
    classification-script classification-script;
    congestion-point-profile-script congestion-point-profile-script;
    extension-path extension-path;
}
```

To configure scripts and classification:

1. From configuration mode, access the configuration statement that configures SRC-ACP scripts and classification. In this sample procedure, the properties are configured in the config group.

```
user@host# edit shared acp group config configuration scripts-and-classification
```

2. Specify the script factory class name.

```
[edit shared acp group config configuration scripts-and-classification]
user@host# set script-factory-class script-factory-class
```

3. Specify the congestion point classifier factory class name.

```
[edit shared acp group config configuration scripts-and-classification]
user@host# set classification-factory-class classification-factory-class
```

4. Specify the class name for congestion point classification.

```
[edit shared acp group config configuration scripts-and-classification]
user@host# set classification-script classification-script
```

5. Specify the class name for generating the congestion point DN by using the congestion point profile.

```
[edit shared acp group config configuration scripts-and-classification]
user@host# set congestion-point-profile-script congestion-point-profile-script
```

6. Specify the extension class path for classes not located in the */opt/UMC/acp/lib* directory.

```
[edit shared acp group config configuration scripts-and-classification]
user@host# set extension-path extension-path
```

7. (Optional) Verify your configuration.

```
[edit shared acp group config configuration scripts-and-classification]
user@host# show
```

Configuring SRC-ACP to Manage the Edge Network

To configure SRC-ACP to manage the edge network you must:

1. Configure network interfaces that represent locations of congestion points in the directory.
2. Configure guaranteed bandwidths for subscribers.
3. Assign network interfaces to subscribers.
4. Configure guaranteed bandwidths for services.

Configuring Network Interfaces in the Directory for the Edge Network

You must add network interfaces to the directory. For the edge network, you do so by specifying the network interfaces of the routers and the switches in the access network between subscribers and the SRC network.

Use the following configuration statements to configure a network interface:

```
shared admission-control device name {
    description description;
}

shared admission-control device name interface name {
    description description;
    upstream-provisioned-rate upstream-provisioned-rate;
    downstream-provisioned-rate downstream-provisioned-rate;
    upstream-background-bandwidth upstream-background-bandwidth;
    downstream-background-bandwidth downstream-background-bandwidth;
}
```

To configure the network interfaces of the routers and the switches in the access network:

1. From configuration mode, access the configuration statement that configures network interfaces.

   ```
   user@host# edit shared admission-control device name
   ```

   Enter the name of the network device.

2. (Optional) Specify a description for the network device.

```
[edit shared admission-control device name]
user@host# set description description
```

3. Specify the network interface.

```
user@host# edit shared admission-control device name interface name
```

   Enter the name of the virtual router.

4. (Optional) Specify the provisioned bandwidth for the network interface.

```
[edit shared admission-control device name interface name]
user@host# set upstream-provisioned-rate upstream-provisioned-rate
user@host# set downstream-provisioned-rate downstream-provisioned-rate
```

5. (Optional) Specify the background bandwidth for the network interface.

```
[edit shared admission-control device name interface name]
user@host# set upstream-background-bandwidth upstream-background-bandwidth
user@host# set downstream-background-bandwidth
downstream-background-bandwidth
```

   For information about background bandwidths, see *Allocating Bandwidth to Applications Not Controlled by SRC-ACP* on page 277.

6. (Optional) Verify your configuration.

```
[edit shared admission-control device name interface name]
user@host# show
```

Configuring Bandwidths for Subscribers

You must configure bandwidths for subscribers that SRC-ACP manages in the edge region of the network.

If the access network between the subscriber and the router uses ATM, and all the traffic coming from one DSLAM travels on a single virtual path, you do not need to provision bandwidths for each subscriber. In this case, SRC-ACP can derive the congestion points from the router (see *Deriving Edge Congestion Points* on page 275.)

However, if the access network uses a protocol other than ATM, you must provide the following information for each subscriber.

- Provisioned downstream bandwidth
- Provisioned upstream bandwidth
- Actual downstream bandwidth for the current subscriber session
- Actual upstream bandwidth for the current subscriber session
- List of DNs of interfaces associated with congestion points

To configure bandwidths for subscribers:

1. From configuration mode, access the configuration statement that configures residential subscribers.

   ```
   user@host# edit subscribers retailer name subscriber-folder folder-name
   subscriber name admission-control
   ```

   For more information about configuring residential subscribers, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 12, Configuring Subscribers and Subscriptions with the SRC CLI*.

2. (Optional) Specify the provisioned downstream bandwidth. This rate is used if the subscriber bandwidth settings are not provided by remote update (through the API for ACP) or by the downstream-sync-rate value.

   ```
   [edit subscribers retailer name subscriber-folder folder-name subscriber name
   admission-control]
   user@host# set downstream-provisioned-rate downstream-provisioned-rate
   ```

3. (Optional) Specify the provisioned upstream bandwidth. This rate is used if the subscriber bandwidth settings are not provided by remote update (through the API for ACP) or by the upstream-sync-rate value.

   ```
   [edit subscribers retailer name subscriber-folder folder-name subscriber name
   admission-control]
   user@host# set upstream-provisioned-rate upstream-provisioned-rate
   ```

4. (Optional) Specify the actual downstream bandwidth for the current subscriber session. If you do not set this value and it is not provided by remote update (through the API for ACP), then the downstream-provisioned-rate value is used.

   ```
   [edit subscribers retailer name subscriber-folder folder-name subscriber name
   admission-control]
   user@host# set downstream-sync-rate downstream-sync-rate
   ```

5. (Optional) Specify the actual upstream bandwidth for the current subscriber session. If you do not set this value and it is not provided by remote update (through the API for ACP), then the upstream-provisioned-rate value is used.

   ```
   [edit subscribers retailer name subscriber-folder folder-name subscriber name
   admission-control]
   user@host# set upstream-sync-rate upstream-sync-rate
   ```

Assigning Network Interfaces to Subscribers

You must assign to the subscriber object interfaces (including the router interfaces) for all congestion points between the subscriber and the router.

NOTE: You must define the interface in the directory before you can assign it to a residential subscriber (see *Configuring Network Interfaces in the Directory for the Edge Network* on page 306).

To assign an interface:

1. From configuration mode, access the configuration statement that configures residential subscribers.

   ```
   user@host# edit subscribers retailer name subscriber-folder folder-name
   subscriber name admission-control
   ```

   For more information about configuring residential subscribers, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 12, Configuring Subscribers and Subscriptions with the SRC CLI*.

2. (Optional) Specify the DNs of interfaces associated with congestion points for this subscriber.

   ```
   [edit subscribers retailer name subscriber-folder folder-name subscriber name
   admission-control]
   user@host# set congestion-points [congestion-points...]
   ```

Configuring Bandwidths for Services in the Edge Network

Upstream and downstream bandwidths must be specified for services that SRC-ACP manages. You can obtain bandwidths for services in two ways:

- Provide static values through the directory.
- Allow the values to be provided through the SAE core API.

  For example, a business partner may need to specify the required values for a particular piece of content through the SAE core API.

To configure values for services:

1. From configuration mode, access the configuration statement that configures services.

   ```
   user@host# edit services global sae-service name admission-control
   ```

   For more information about configuring services, see *SRC-PE Services and Policies Guide, Chapter 1, Managing Services with the SRC CLI*.

2. (Optional) Specify the required downstream and upstream bandwidths.

   ```
   [edit services global sae-service name admission-control]
   user@host# set required-downstream-bandwidth required-downstream-bandwidth
   user@host# set required-upstream-bandwidth required-upstream-bandwidth
   ```

Configuring SRC-ACP to Manage the Backbone Network

To configure SRC-ACP to manage the backbone network, you must:

1. Configure network interfaces that represent locations of congestion points in the directory.
2. (Optional) Configure an action congestion point.
3. Configure guaranteed bandwidths for services.
4. Assign network interfaces to services.
5. Create congestion points in the directory.
6. Assign network interfaces to congestion points.

Configuring Network Interfaces in the Directory for the Backbone Network

You configure network interfaces in the directory in the same way for edge and backbone congestion points. However, for backbone congestion points, you can add only VRs and their interfaces. For information about this procedure, see *Configuring Network Interfaces in the Directory for the Edge Network* on page 306.

Extending SRC-ACP Congestion Points for the Backbone Network

You can extend SRC-ACP congestion points to initialize and execute applications defined in a backbone congestion point. SRC-ACP provides a service provider interface (SPI) to:

- Create custom congestion point applications that authorize service activation and track service start and stop events.
- Obtain congestion point information from remote update.
- Retrieve congestion point status.
- Track congestion point state.

The SPI for ACP provides a Java interface that a congestion point application implements. For information about the SPI for ACP, see the documentation in the SRC application library distribution in the folder *SDK/doc/acp*.

The implementation of the SPI for ACP can be a customized application that performs certain tasks, such as creating or removing congestion points on the router. SRC-ACP acts as an interface tracking plug-in, and interface tracking events are treated as remote updates for congestion points when they are created, modified, or removed.

SRC-ACP supports applications written in Java or Jython. For scripts written in Java, you must compile and package the implemented SPI for ACP to make it available for use by SRC-ACP. A Java implementation can include more than one Java archive (JAR) file.

To use congestion point applications with SRC-ACP, configure an action congestion point that references the script (see *Configuring Action Congestion Points* on page 311).

Configuring Action Congestion Points

You can define an application in a backbone congestion point so that SRC-ACP can execute it in a predefined manner. Backbone congestion points that are configured to run an application are called action congestion points. If you want to use an action congestion point to execute an application that requires real-time congestion point status, you must enable SRC-ACP state synchronization with the SAE (see *Configuring SRC-ACP Redundancy* on page 300).

Before you configure an action congestion point, make sure that you know the location of the application file.

Use the following configuration statements to configure action congestion points:

```
shared admission-control device name interface name {
    action-type (url | python | java-class | java-archive);
    action-class-name action-class-name;
    action-file-url action-file-url;
    action-parameters [action-parameters...];
}
```

To configure an action congestion point:

1. From configuration mode, access the configuration statement that configures network interfaces.

   ```
   user@host# edit shared admission-control device name interface name
   ```

   Enter the name of the network device and the name of the virtual router.

2. (Optional) Specify the file type of the application.

   ```
   [edit shared admission-control device name interface name]
   user@host# set action-type (url | python | java-class | java-archive);
   ```

3. (Optional) Specify the name of the class implementing the SPI.

   ```
   [edit shared admission-control device name interface name]
   user@host# set action-class-name action-class-name
   ```

4. (Optional) Specify the URL or the content of the script file.

   ```
   [edit shared admission-control device name interface name]
   user@host# set action-file-url action-file-url
   ```

5. (Optional) Specify the parameter as an attribute = value pair.

   ```
   [edit shared admission-control device name interface name]
   user@host# set action-parameters [action-parameters...]
   ```

6. (Optional) Verify your configuration.

```
[edit shared admission-control device name interface name]
user@host# show
```

Configuring Bandwidths for Services in the Backbone Network

You configure bandwidths for services in the same way for edge and backbone congestion points. For information about this procedure, see *Configuring Bandwidths for Services in the Edge Network* on page 309.

Configuring Congestion Points for Services in the Backbone Network

You must assign a congestion point to each service that SRC-ACP manages.

To configure values for services:

1. From configuration mode, access the configuration statement that configures services.

   ```
   user@host# edit services global sae-service name admission-control
   ```

   For more information about configuring services, see *SRC-PE Services and Policies Guide, Chapter 1, Managing Services with the SRC CLI*.

2. (Optional) Specify the backbone congestion points.

   ```
   [edit services global sae-service name admission-control]
   user@host# set congestion-points [congestion-points...]
   ```

   The backbone congestion point is defined in the format <-vrName->/<-serviceName->.

   - To allow the software to automatically define the congestion point, use the entry <-vrName->/<-serviceName->. When SRC-ACP starts operating, it will substitute the VR name and the service name from the request for service activation.

   - To restrict the congestion point to a specific VR or service, enter the actual VR name or service name.

Configuring Congestion Points in the Directory

To configure individual backbone congestion points:

1. From configuration mode, access the configuration statement that configures congestion points.

   ```
   user@host# edit shared congestion-points profile name
   ```

   Enter the name of the virtual router that supports the congestion point.

2. (Optional) Verify your configuration.

   ```
   [edit shared congestion-points profile name]
   user@host# show
   ```

Assigning Interfaces to Congestion Points

You must assign interfaces either to VRs or to individual services under the VRs. Services inherit interface assignments from the associated VR unless you assign an interface to the individual service. This network interface lists the DNs of interfaces associated with backbone congestion points.

Use the following configuration statements to configure interface assignments:

```
shared congestion-points profile name {
    interface [interface...];
}
```

To assign interfaces to congestion points:

1. From configuration mode, access the configuration statement that configures congestion points.

   ```
   user@host# edit shared congestion-points profile name
   ```

   Enter the name of the network device to which you want to assign the congestion point.

2. (Optional) Specify the interfaces associated with a congestion point for this subscriber.

   ```
   [edit shared congestion-points profile name]
   user@host# set interface interface
   ```

3. (Optional) Verify your configuration.

   ```
   [edit shared congestion-points profile name]
   user@host# show
   ```

Chapter 22

Configuring Congestion Point Classification with the SRC CLI

This chapter describes how to use the SRC command-line interface (CLI) to configure congestion point classification in the SRC Admission Control Plug-In (SRC-ACP) application. You can use the CLI to configure SRC-ACP on a Solaris platform or on a C-series Controller.

You can also use SRC configuration applications to configure congestion point classification on a Solaris platform. See *Chapter 26, Providing Admission Control with SRC-ACP on a Solaris Platform*.

Topics in this chapter include:

- Overview of Congestion Point Classification on page 315
- Configuration Statements for Congestion Point Classification on page 316
- Classifying Congestion Points on page 316
- *Defining a Congestion Point Profile* on page 321

Overview of Congestion Point Classification

Congestion point classification allows you to automate and scale the configuration of congestion points. SRC-ACP uses classification scripts to determine which congestion point to load for a subscriber. SRC-ACP can select the congestion point from congestion point profiles or subscriber profiles.

Congestion Point Classification Scripts

The congestion point classification scripts consist of targets and criteria.

- A target is the result of the classification script. The result of congestion point classification scripts is an LDAP search string that is used to find a unique congestion point in the directory. If no classification scripts are configured, the result of congestion point classification scripts is an LDAP search string for the subscriber profile of the particular subscriber.

- Criteria are match criteria. The script attempts to match criteria in the script to information sent from the router. Match criteria for a congestion point classification script might be a subscriber distinguished name (DN) or an interface name.

Each script can have multiple targets, and each target can have multiple criteria. When an object needs classification, the script processes the targets in turn. Within each target, the script processes criteria sequentially. When it finds that the classification criteria for a target match, it returns the target to SRC-ACP.

Because classification scripts examine criteria sequentially as the criteria appear in the script, you should put more specific criteria at the beginning of the script and less specific criteria at the end of the script.

Congestion Point Profiles

Congestion point profiles are used to share congestion points that are generated based on dynamic configuration information. SRC-ACP uses congestion point profiles to determine the set of congestion points based on the classification script results.

Configuration Statements for Congestion Point Classification

Use the following configuration statements to configure congestion point classification at the [edit] hierarchy level.

```
shared acp congestion-point-classifier rule name {
    target target;
    script script;
}

shared acp congestion-point-classifier rule name condition name ...

shared congestion-points congestion-point-profile name {
    expression [expression...];
}
```

For detailed information about each configuration statement, see the *SRC-PE CLI Command Reference.*

Classifying Congestion Points

Use the following configuration statements to configure congestion point classification scripts:

```
shared acp congestion-point-classifier rule name {
    target target;
    script script;
}

shared acp congestion-point-classifier rule name condition name ...
```

Changes that you make to classification scripts do not affect subscriber sessions that are already established.

Configuring Targets and Criteria for Classification Scripts

To define a target and criteria for the congestion point classification script:

1. From configuration mode, access the configuration statement that configures congestion point scripts. In this sample procedure, the scripts are configured in the config group.

   ```
   user@host# edit shared acp group config congestion-point-classifier rule name
   ```

   Enter a name for the congestion point classification script.

2. Specify the target for the classification script.

   ```
   [edit shared acp group config congestion-point-classifier rule name]
   user@host# set target target
   ```

   For information about classification targets, see *Configuring Congestion Point Classification Targets* on page 318.

3. Specify the classification criteria for the target.

   ```
   [edit shared acp group config congestion-point-classifier rule name]
   user@host# set script script
   ```

   For information about classification criteria, see *Selecting Congestion Point Classification Criteria* on page 318.

Configuring Classification Scripts Contents for Classification Scripts

To use the contents of a classification script to another object for the congestion point classification script:

- Access the configuration statement that configures congestion point scripts from configuration mode. In this sample procedure, the scripts are configured in the config group.

  ```
  user@host# edit shared acp group config congestion-point-classifier rule name
  condition name ...
  ```

  Enter a name for the congestion point classification script and the name of the classification script that you want to use.

Configuring Congestion Point Classification Targets

The target of the congestion point classification script is an LDAP search string. The search string uses a syntax similar to an LDAP URL (see RFC 2255—The LDAP URL Format (December 1997)). The syntax is:

```
baseDN [ ? [ attributes ] [ ? [ scope ] [ ? [ filter ] ] ] ]
```

- baseDN—Distinguished name (DN) of the object where the LDAP search starts.
- attributes—Is ignored.
- scope—Scope of search in the directory:
  - base—Default; searches the base DN only.
  - one—Searches the direct children of the base DN.
  - sub—Searches the complete subtree below the base DN.
- filter—An RFC 2254–style LDAP search filter expression; for example, (uniqueId=<-userName->). See RFC 2254—The String Representation of LDAP Search Filters (December 1997).

With the exception of baseDN all the fields are optional.

The result of the LDAP search must be exactly one directory object. If no object or more than one object is found, congestion points for the subscriber are not loaded and all service activations for the subscriber are denied.

Selecting Congestion Point Classification Criteria

Congestion point classification criteria define match criteria that are used to find the congestion point profile. Use the fields in this section to define classification criteria.

accountingId

- Value of directory attribute accountingUserId.

authUserId

- Identifier that a subscriber uses for authentication.
- Value—Username

dhcpPacket

- Content of the DHCP discover request.
- Value—Byte array
  - First 4 octets—Gateway IP address (giaddr field)
  - Remaining octets—DHCP options

For more information, see RFC 2131—Dynamic Host Configuration Protocol (March 1997) and RFC 2132—DHCP Options and BOOTP Vendor Extensions (March 1997).

domain

- Name of the domain used for secondary authentication.
- Value—Valid domain name
- Example—domain = "isp99.com"

ifRadiusClass

- RADIUS class attribute on the JUNOSe interface.
- Value—RADIUS class name
- Example—ifRadiusClass = "acpe"

ifSessionId

- Identifier for RADIUS accounting on the JUNOSe interface.

interfaceAlias

- Description of the interface.
- Value—Interface description that is configured on the JUNOSe router with the interface ip description command
- Example—interfaceAlias = "dhcp-subscriber12"

interfaceDescr

- Alternate name for the interface that is used by SNMP. This name is a system-generated name.
- Value
  - On a JUNOSe router, the format of the description is

    ip<slot>/<port>.<subinterface>
  - On the JUNOS routing platform, interfaceDescr is the same as interfaceName.
- Example—interfaceDescr = "IP3/1"

interfaceName

- Name of the interface.
- Value
  - Name of the interface in your router CLI syntax
  - FORWARDING_INTERFACE for routing instance (used by traffic mirroring)
- Example—For JUNOSe routers: interfaceName = "fastEthernet6/0"

  For JUNOS routing platforms: interfaceName = "fe-0/1/0.0"

  For forwarding interface: interfaceName = "FORWARDING_INTERFACE"

loginName

- Subscriber's login name.
- Value—Login name
- Guidelines—The format of the login name varies. A loginName can be of form subscriber, domain\subscriber, subscriber@domain, or as otherwise defined by the login setup of the manager.
- Example—idp@idp

nasIp

- IP address of the router.
- Value—Byte array
  - For IPv4 address—4 octets in network byte order
  - For IPv6 address—16 octets in network byte order

nasPort

- Port identifier of an interface.
- Value—Includes interface name and additional layer 2 information
- Example—nasPort = "fastEthernet 3/1" (There is a space between fastEthernet and slot number 3/1 in the nasPort field.)

portId

- Identifier of VLAN or virtual circuit.
- Value—String; for a virtual circuit, use the format <VPI>/<VCI>

primaryUserName

- PPP login name or the public DHCP username.
- Value—Subscriber name
- Example—primaryUserName = "peter"

radiusClass

- RADIUS class attribute of the service definition.
- Value—RADIUS class name
- Example—radiusClass = "Premium"

routerName

- Name of virtual router.
- Value—Virtual router name in the format <virtualRouter>@<router>
- Example—routerName = "default@e_series5"

sessionId

- Identifier of RADIUS session for the subscriber session.

serviceBundle

- Content of the RADIUS vendor-specific attribute for the service bundle.
- Value—Name of a service bundle
- Example—serviceBundle = “goldSubscriber”

sspHost

- Name of host on which the SAE is installed.

userDn

- DN of a subscriber in the directory.
- Value—DN of a subscriber profile

userIp

- IP address of the subscriber.
- Value—Byte array
  - For IPv4 address—4 octets in network byte order
  - For IPv6 address—16 octets in network byte order

userMacAddress

- Media access control (MAC) address of the DHCP subscriber.
- Value—Valid MAC address
- Example—userMacAddress = “00:11:22:33:44:55”

userType

- Type of subscriber.

Defining a Congestion Point Profile

You can create a congestion point profile that automatically performs congestion point classification. This profile supports only access network mode for SRC-ACP.

Use the following configuration statements to configure congestion point profiles:

```
shared congestion-points congestion-point-profile name {
    expression [expression...];
}
```

To define a congestion point profile:

1. From configuration mode, access the configuration statement that configures congestion point profiles.

   ```
   user@host# edit shared congestion-points congestion-point-profile name
   ```

   Enter a name for the profile.

2. Specify congestion point expressions.

   ```
   [edit shared congestion-points congestion-point-profile name]
   user@host# set expression [expression...]
   ```

   For information about congestion point expressions, see *Congestion Point Expressions* on page 322.

Congestion Point Expressions

You can enter a congestion point expression by using the syntax listed in this section. You can also embed Python scripting expressions within the congestion point expression.

If you embed Python expressions within a congestion point expression, use the escape sequence <- then -> to enclose the Python expression. See *Methods for Use with Scripting Expressions* on page 323 and *Match Criteria for Congestion Point Classification* on page 323.

The syntax for a congestion point expression is:

<NetworkDevice>/<NetworkInterface>[/<CongestionPoint>]

- <NetworkDevice>—Network device listed in the directory.

  For information about network devices, see *SRC-PE Network Guide, Part 2, Using Juniper Networks Routers in the SRC Network*.

- <NetworkInterface>—Network interface listed in the directory.

  For information about interfaces, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 6, Classifying Interfaces and Subscribers with the SRC CLI*.

- <CongestionPoint>—(Optional) Name of an instance of a congestion point that is automatically created.

If one of the elements with the path contains a slash (/), use a backslash (\) as an escape character for the slash. For example, \/.

Expressions in Templates for Congestion Point Profiles

You can create a congestion point profile to be used as a template for other profiles. Templates simplify management of congestion points. Rather than configuring each congestion point individually, you can create templates to define common parameters for a class of individual congestion points.

For example, in an environment in which VLAN interfaces GigabitEthernet1/0.1 through GigabitEthernet1/0.1000 have the same available bandwidth, you can specify the characteristics of the VLAN interface once and have SRC-ACP create the congestion points based on the template configuration.

When a congestion point expression has the third element (<CongestionPoint>), SRC-ACP uses the <NetworkDevice>/<NetworkInterface> part of the expression to load the congestion point from the directory, and uses it as a template to create a congestion point in memory for subscriber. The <CongestionPoint> part of the expression distinguishes each congestion point (available bandwidth) created from this template.

Methods for Use with Scripting Expressions

SRC-ACP provides the following methods to use in scripting expressions:

- slot(nasPortId)—Collects the slot number from the nasPortId or interfaceName

  Example—slot("atm 4/5:0.32") == "4"

- port(nasPortId)—Collects the port number from the nasPortId or interfaceName

  Example—port("atm 4/5:0.32") == "5"

- l2id(nasPortId)—Collects the layer 2 ID from the nasPortId (VLAN id or ATM vpi.vci)

  Example—l2id("atm 4/5:0.32") == "0.32"

- escape(string)—Replaces any slash with the escape sequence \/

  Example—escape("atm 4/5") == "atm 4\/5"

Match Criteria for Congestion Point Classification

You can use the match criteria in Python scripting expressions for a congestion point expression. For more information about the match criteria, see *Selecting Congestion Point Classification Criteria* on page 318.

Chapter 23

Managing SRC-ACP with the SRC CLI

This chapter describes how to use the SRC command-line interface (CLI) to manage the SRC Admission Control Plug-In (SRC-ACP) application. You can use the CLI to manage SRC-ACP on a Solaris platform or on a C-series Controller.

Topics in this chapter include:

Starting SRC-ACP

To start SRC-ACP:

```
user@host> enable component acp
```

Stopping SRC-ACP

To stop SRC-ACP:

```
user@host> disable component acp
```

Reorganizing the File That Contains ACP Data

Periodically, you should reorganize the files that contain ACP data about subscribers, services, and congestion points. This action reduces the sizes of these files. To do so:

```
user@host> request acp reorganize-backup-database
```

Modifying Congestion Points

By default, SRC-ACP does not register changes in congestion points until you stop and restart SRC-ACP. To modify the congestion point associated with a service without stopping and starting SRC-ACP:

1. Make sure that no subscribers have subscriptions to services that use the congestion point you want to modify.
2. From configuration mode, access the configuration statement that configures SRC-ACP connections to the services' directory.

   ```
   user@host# edit shared acp configuration ldap service-data
   ```

3. Specify whether SRC-ACP detects changes in the backbone congestion point for a service while SRC-ACP is operative.

   ```
   [edit shared acp configuration ldap service-data]
   user@host# set reload-congestion-points
   ```

4. Wait for 30 seconds before you proceed to the next step.

   Depending on the value of the polling interval for directory eventing, SRC-ACP may take up to 30 seconds to register the change to the reload-congestion-points option. If you modify the congestion point before SRC-ACP registers the new setting for the reload-congestion-points option, SRC-ACP will not register the change for the congestion point.

5. Modify the congestion point in the service definition. See *Configuring Congestion Points for Services in the Backbone Network* on page 312.

   SRC-ACP immediately registers the change.

6. From configuration mode, access the configuration statement that configures SRC-ACP connections to the services' directory.

   ```
   user@host# edit shared acp configuration ldap service-data
   ```

7. Specify whether SRC-ACP detects changes in the backbone congestion point for a service while SRC-ACP is operative.

   ```
   [edit shared acp configuration ldap service-data]
   user@host# set reload-congestion-points
   ```

Chapter 24

Monitoring Admission Control with the SRC CLI

This chapter describes how to use the SRC command-line interface (CLI) to monitor information about the SRC Admission Control Plug-In (SRC-ACP) application. You can use the CLI to monitor admission control on a Solaris platform or on a C-series Controller.

You can also use the C-Web interface to monitor admission control. See *Chapter 25, Monitoring Admission Control with the C-Web Interface*.

Topics in this chapter include:

Viewing Information About Subscriber Sessions in the Edge Network

To display information about the current subscriber sessions in memory:

```
user@host> show acp edge subscriber
```

To display information about specific subscriber sessions:

```
user@host> show acp edge subscriber session-id session-id
```

Enter all or part of the subscriber session ID to list all matching subscriber sessions.

To display information about the subscriber sessions from a specific virtual router:

```
user@host> show acp edge subscriber virtual-router-name virtual-router-name
```

Enter a virtual router name to list subscriber sessions from a particular virtual router.

To display subscriber session attributes for the current subscriber sessions:

```
user@host> show acp edge subscriber brief
```

By default, information about the subscriber session attributes, service sessions, and associated congestion points is displayed.

Viewing Information About Congestion Points in the Edge Network

You can display information about edge congestion points by distinguished name (DN) or by subscriber session.

Viewing Edge Congestion Point Information by DN

To display information about edge congestion points by DN:

```
user@host> show acp edge congestion-point dn
```

To display information about specific congestion points by DN:

```
user@host> show acp edge congestion-point dn congestion-point-dn
congestion-point-dn
```

Enter a partial congestion point DN to list all matching congestion points.

To display information about specific congestion points that were generated dynamically by instance ID:

```
user@host> show acp edge congestion-point dn instance-id instance-id
user@host> show acp edge congestion-point dn congestion-point-dn
congestion-point-dn instance-id instance-id
```

When a congestion point is dynamically generated with a congestion point profile, the generated instance ID is appended to the congestion point DN. Enter a partial instance ID to list all matching congestion points.

To display information about the congestion points from a specific virtual router:

```
user@host> show acp edge congestion-point dn virtual-router-name
virtual-router-name
```

Enter a virtual router name to list congestion points from a particular virtual router.

To display congestion point DNs:

```
user@host> show acp edge congestion-point dn brief
```

By default, information about the congestion point attributes and congestion point bandwidth usage is displayed.

To restrict the number of displayed results:

```
user@host> show acp edge congestion-point dn maximum-results maximum-results
```

Viewing Edge Congestion Point Information by Subscriber Session

To display information about edge congestion points by subscriber session:

```
user@host> show acp edge congestion-point subscriber-session-id
```

To display information about specific congestion points by subscriber session:

```
user@host> show acp edge congestion-point subscriber-session-id session-id
session-id
```

Enter a partial subscriber session ID to list all matching congestion points.

To display information about the congestion points from a specific virtual router:

```
user@host> show acp edge congestion-point subscriber-session-id
virtual-router-name virtual-router-name
```

Enter a virtual router name to list congestion points from a particular virtual router.

To display congestion point DNs:

```
user@host> show acp edge congestion-point subscriber-session-id brief
```

By default, information about the congestion point attributes and congestion point bandwidth is displayed.

To restrict the number of displayed results:

```
user@host> show acp edge congestion-point subscriber-session-id maximum-results
maximum-results
```

Viewing Information About Services in the Backbone Network

To display information about services that SRC-ACP manages in the backbone network:

user@host> **show acp backbone service**

To display information about specific backbone service used to generate congestion points:

user@host> **show acp backbone service service-name** *service-name*

Enter a partial service name to list all matching backbone services.

To display information about the backbone services from a specific virtual router:

user@host> **show acp backbone service virtual-router-name** *virtual-router-name*

Enter a virtual router name to list backbone services from a particular virtual router.

To display backbone service attributes:

user@host> **show acp backbone service brief**

By default, information about the backbone service attributes, service sessions, and associated congestion points is displayed.

Viewing Information About Congestion Points in the Backbone Network

The procedure for displaying information about backbone congestion points is similar to the procedure for displaying information about edge congestion points. However, you have the option of specifying a service, whereas in the edge network you have the option of specifying an identifier for a subscriber session.

You can display information about backbone congestion points by DN or by service.

Viewing Backbone Congestion Point Information by DN

To display information about backbone congestion points by DN:

user@host> **show acp backbone congestion-point dn**

To display information about specific congestion points by DN:

user@host> **show acp backbone congestion-point dn congestion-point-dn** *congestion-point-dn*

Enter a partial congestion point DN to list all matching congestion points.

To display information about the congestion points from a specific virtual router:

user@host> **show acp backbone congestion-point dn virtual-router-name** *virtual-router-name*

Enter a virtual router name to list congestion points from a particular virtual router.

To display congestion point DNs:

user@host> **show acp backbone congestion-point dn brief**

By default, information about the congestion point attributes and congestion point bandwidth usage is displayed.

Viewing Backbone Congestion Point Information by Service

To display information about backbone congestion points by service:

user@host> **show acp backbone congestion-point congestion-point-expression**

To display information about specific backbone services used to generate congestion points:

user@host> **show acp backbone congestion-point congestion-point-expression service-name** *service-name*

Enter a partial service name to list all matching backbone services.

To display information about the backbone services from a specific virtual router:

user@host> **show acp backbone congestion-point congestion-point-expression virtual-router-name** *virtual-router-name*

Enter a virtual router name to list backbone services from a particular virtual router.

To display congestion point DNs:

user@host> **show acp backbone congestion-point congestion-point-expression brief**

By default, information about the congestion point attributes and congestion point bandwidth is displayed.

Viewing Information About Action Congestion Points in the Backbone Network

Backbone congestion points that are configured to run applications are action congestion points. You can display information about action congestion points by service or by congestion point.

Viewing Action Congestion Point Information by Service

To display information about services that SRC-ACP manages in the backbone network:

user@host> **show acp backbone service**

To display information about specific backbone services used to generate congestion points:

user@host> **show acp backbone service service-name** *service-name*

Enter a partial service name to list all matching backbone services.

To display information about the backbone services from a specific virtual router:

```
user@host> show acp backbone service virtual-router-name virtual-router-name
```

Enter a virtual router name to list backbone services from a particular virtual router.

To display backbone service attributes:

```
user@host> show acp backbone service brief
```

By default, information about the backbone service attributes, service sessions, and associated congestion points is displayed.

Viewing Action Congestion Point Information by Congestion Point

To display information about backbone congestion points by service:

```
user@host> show acp backbone congestion-point congestion-point-expression
```

To display information about specific backbone services used to generate congestion points:

```
user@host> show acp backbone congestion-point congestion-point-expression
service-name service-name
```

Enter a partial service name to list all matching backbone services.

To display information about the backbone services from a specific virtual router:

```
user@host> show acp backbone congestion-point congestion-point-expression
virtual-router-name virtual-router-name
```

Enter a virtual router name to list backbone services from a particular virtual router.

To display information about the backbone services from a specific interface:

```
user@host> show acp backbone congestion-point congestion-point-expression
interface-name interface-name
```

Enter an interface name to list backbone services from a particular interface.

To display information about the backbone services for a specific interface description:

```
user@host> show acp backbone congestion-point congestion-point-expression
interface-description interface-description
```

Enter an interface description to list backbone services for a particular description.

To display information about the backbone services from a specific interface alias:

```
user@host> show acp backbone congestion-point congestion-point-expression
interface-alias interface-alias
```

Enter an interface alias to list backbone services from a particular alias.

To display information about the backbone services for a specific NAS port ID:

```
user@host> show acp backbone congestion-point congestion-point-expression
nasPort-id nasPort-id
```

Enter a NAS port ID to list backbone services from a particular ID.

To display congestion point DNs:

```
user@host> show acp backbone congestion-point congestion-point-expression brief
```

By default, information about the congestion point attributes and congestion point bandwidth is displayed.

Viewing Information About Subscribers Obtained from External Applications

To display information about subscribers added through an external application:

```
user@host> show acp remote-update subscriber
```

To display information about subscribers connected from a specific device:

```
user@host> show acp remote-update subscriber device-name device-name
```

Enter a device name to list subscribers connected from a particular device.

To display information about specific subscribers connected from a specific interface:

```
user@host> show acp remote-update subscriber nas-port-id nas-port-id
```

Enter the NAS port ID of interface to list all matching subscribers connected from a particular interface.

To display information about specific subscribers connected from a specific NAS IP address:

```
user@host> show acp remote-update subscriber nas-ip nas-ip
```

Enter the NAS IP address of the device to list all matching subscribers connected from a particular device.

To display information about specific subscribers connected from a specific subscriber IP address:

```
user@host> show acp remote-update subscriber subscriber-ip subscriber-ip
```

Enter the subscriber IP address to list all matching subscribers connected from a particular address.

To display information about the subscribers from a specific phone number:

```
user@host> show acp remote-update subscriber phone phone
```

Enter a phone number to list subscribers from a particular phone number.

To display subscriber attributes:

user@host> **show acp remote-update subscriber brief**

By default, information about the subscriber attributes, service sessions, and associated congestion points is displayed.

Viewing Information About Congestion Points Added Through an External Application

You can display information about congestion points added through an external application by DN or by interface name.

Viewing Congestion Point Information by DN

To display information about congestion points added through an external application by DN:

user@host> **show acp remote-update congestion-point dn**

To display information about specific congestion points by DN:

user@host> **show acp remote-update congestion-point dn congestion-point-dn** *congestion-point-dn*

Enter a partial congestion point DN to list all matching congestion points.

To display congestion point DNs:

user@host> **show acp remote-update congestion-point dn brief**

By default, information about the congestion point attributes and congestion point bandwidth usage is displayed.

Viewing Congestion Point Information by Name

To display information about congestion points added through an external application by interface name:

user@host> **show acp remote-update congestion-point name**

To display information about congestion points connected from a specific device:

user@host> **show acp remote-update congestion-point name device-name** *device-name*

Enter a device name to list congestion points connected from a particular device.

To display information about specific subscribers connected from a specific interface:

user@host> **show acp remote-update congestion-point name interface-name** *interface-name*

Enter the interface name to list all matching congestion points connected from a particular interface.

To display congestion point DN:

```
user@host> show acp remote-update congestion-point name brief
```

By default, information about the congestion point attributes and congestion point bandwidth usage is displayed.

Viewing SNMP Information for Devices

To display statistics for SNMP information about each device:

```
user@host> show acp statistics device
```

To display statistics for SNMP information about specific devices:

```
user@host> show acp statistics device filter filter
```

Enter a partial device name to list information for all matching devices.

Viewing SNMP Information for the Directory

To display statistics for directory SNMP information:

```
user@host> show acp statistics directory
```

Viewing SNMP Information for SRC-ACP

To display statistics for SRC-ACP SNMP information:

```
user@host> show acp statistics general
```

Chapter 25

Monitoring Admission Control with the C-Web Interface

This chapter describes how to use the C-Web interface to monitor information about the SRC Admission Control Plug-In (SRC-ACP) application. You can use the C-Web interface to monitor admission control on a Solaris platform or on a C-series Controller.

This chapter contains the following topics:

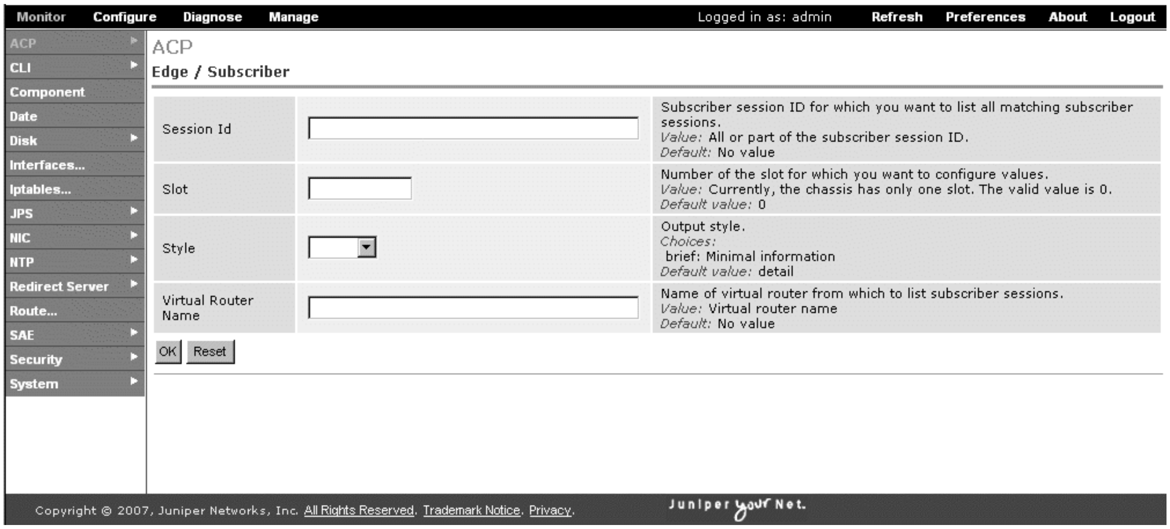
- Viewing Information About Subscriber Sessions in the Edge Network with the C-Web Interface on page 338
- Viewing Information About Congestion Points in the Edge Network with the C-Web Interface on page 339
- Viewing Information About Services in a Backbone Network with the C-Web Interface on page 341
- Viewing Information About Congestion Points in a Backbone Network with the C-Web Interface on page 342
- Viewing Information About Action Congestion Points in a Backbone Network with the C-Web Interface on page 344
- Viewing Information About Subscribers Obtained from External Applications with the C-Web Interface on page 347
- Viewing Information About Congestion Points Added Through an External Application with the C-Web Interface on page 348
- Viewing Statistics for the SRC-ACP Configuration with the C-Web Interface on page 350

Viewing Information About Subscriber Sessions in the Edge Network with the C-Web Interface

To view information about subscriber sessions:

1. Click **ACP > Edge > Subscriber**.

   The Edge/Subscriber pane appears.



Session Id		Subscriber session ID for which you want to list all matching subscriber sessions. *Value:* All or part of the subscriber session ID. *Default:* No value
Slot		Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style		Output style. *Choices:* brief: Minimal information *Default value:* detail
Virtual Router Name		Name of virtual router from which to list subscriber sessions. *Value:* Virtual router name *Default:* No value

2. In the Session ID box, enter a full or partial session ID name to display information about one or more specific sessions, or leave this field empty to display information about all sessions.

3. In the Slot box, enter the number of the slot for which you want to display subscriber session information.

4. Select an output style from the Style list.

5. In the Virtual Router Name box, enter a virtual router name to display information about a specific virtual router, or leave the box empty to display information about all virtual routers.

6. Click **OK**.

   The Edge/Subscriber pane displays a list of current subscriber sessions.

Viewing Information About Congestion Points in the Edge Network with the C-Web Interface

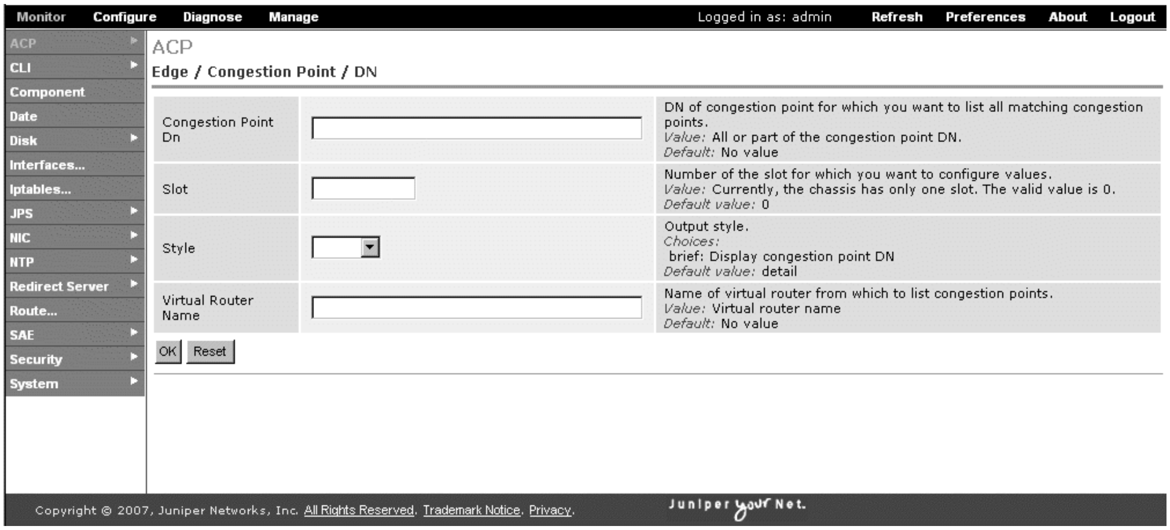
You can display information about edge congestion points by distinguished name (DN) or by subscriber session.

Viewing Information About Edge Congestion Points by DN

To view information about edge congestion points:

1. Click **ACP > Edge > Congestion Point > DN**.

   The Edge/Congestion Point/DN pane appears.



ACP

Edge / Congestion Point / DN

Congestion Point Dn		DN of congestion point for which you want to list all matching congestion points. *Value:* All or part of the congestion point DN. *Default:* No value
Slot		Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style		Output style. *Choices:* brief: Display congestion point DN *Default value:* detail
Virtual Router Name		Name of virtual router from which to list congestion points. *Value:* Virtual router name *Default:* No value

2. In the Congestion Point DN box, enter a congestion point DN, or leave the box blank to view information for all DNs.

3. In the Slot box, enter the number of the slot for which you want to display congestion point information.

4. Select an output style from the Style list.

5. In the Virtual Router Name box, enter a virtual router name to display information about a specific virtual router, or leave the box empty to display information about all virtual routers.

6. Click **OK**.

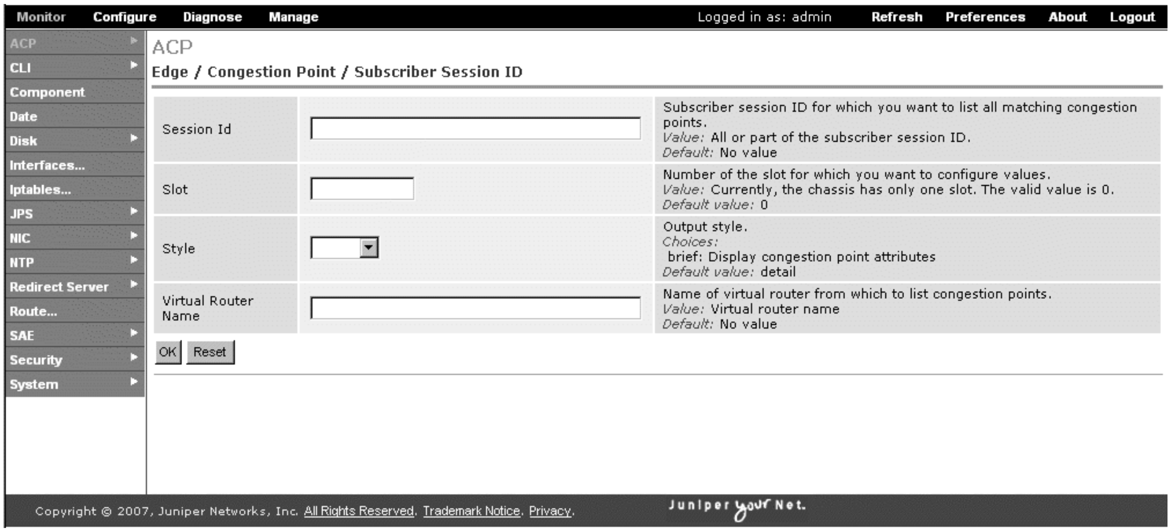
   The Edge/Congestion Point/DN pane displays a list of congestion points.

Viewing Information About Edge Congestion Points by Subscriber Session

To view information about edge congestion points:

1. Click **ACP > Edge > Congestion Point > Subscriber Session ID**.

   The Edge/Congestion Point/Subscriber Session ID pane appears.



ACP

Edge / Congestion Point / Subscriber Session ID

Session Id		Subscriber session ID for which you want to list all matching congestion points. *Value:* All or part of the subscriber session ID. *Default:* No value
Slot		Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style		Output style. *Choices:* brief: Display congestion point attributes *Default value:* detail
Virtual Router Name		Name of virtual router from which to list congestion points. *Value:* Virtual router name *Default:* No value

2. In the Session ID box, enter a full or partial session ID name to display information about one or more specific sessions, or leave the box empty to display information about all sessions.

3. In the Slot box, enter the number of the slot for which you want to display congestion point information.

4. Select an output style from the Style list.

5. In the Virtual Router Name box, enter a virtual router name to display information about a specific virtual router, or leave the box empty to display information about all virtual routers.

6. Click **OK**.

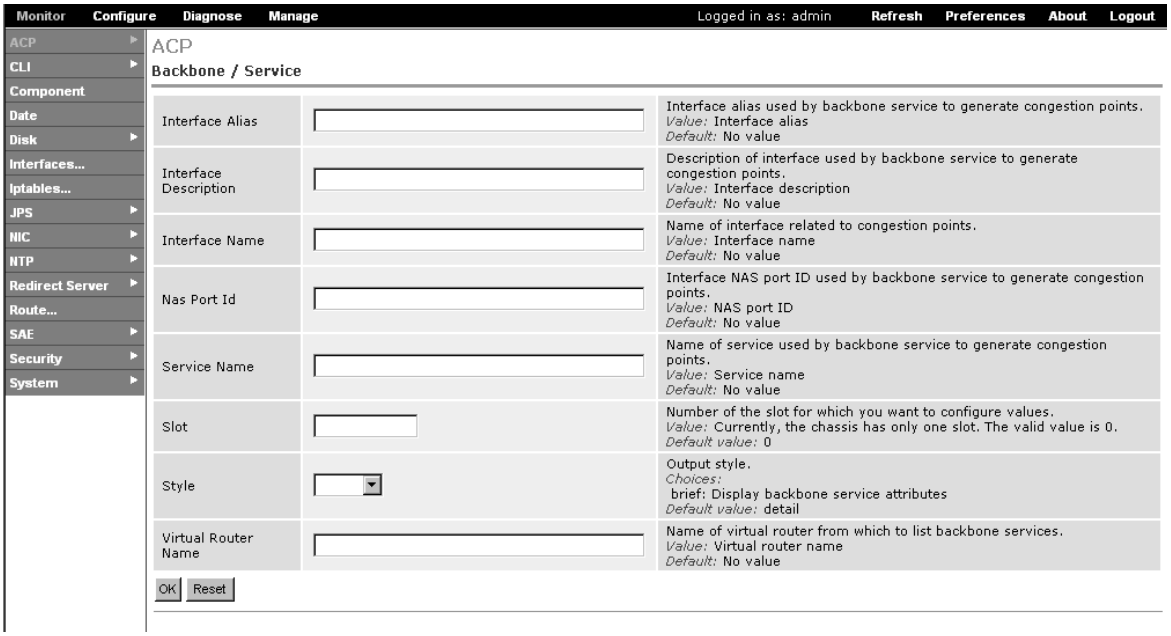
   The Edge/Congestion Point/Subscriber Session ID pane displays a list of congestion points.

Viewing Information About Services in a Backbone Network with the C-Web Interface

To view information about services in a backbone network:

1. Click **ACP > Backbone > Service**.

   The Backbone/Service pane appears.



ACP

Backbone / Service

Interface Alias	Interface alias used by backbone service to generate congestion points. *Value:* Interface alias *Default:* No value
Interface Description	Description of interface used by backbone service to generate congestion points. *Value:* Interface description *Default:* No value
Interface Name	Name of interface related to congestion points. *Value:* Interface name *Default:* No value
Nas Port Id	Interface NAS port ID used by backbone service to generate congestion points. *Value:* NAS port ID *Default:* No value
Service Name	Name of service used by backbone service to generate congestion points. *Value:* Service name *Default:* No value
Slot	Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style	Output style. *Choices:* brief: Display backbone service attributes *Default value:* detail
Virtual Router Name	Name of virtual router from which to list backbone services. *Value:* Virtual router name *Default:* No value

2. In the Interface Alias box, enter the interface alias used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.

3. In the Interface Description box, enter the interface description used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.

4. In the Interface Name box, enter the name of an interface to display information about one interface, or leave the box empty to display information about all interfaces.

5. In the NAS Port ID box, enter the NAS port ID used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.

6. In the Service Name box, enter the name of a service to display information about one service, or leave the box empty to display information about all services.

7. In the Slot box, enter the number of the slot for which you want to display congestion point information.

8. Select an output style from the Style list.

9. In the Virtual Router Name box, enter a virtual router name to display information about a specific virtual router, or leave the box empty to display information about all virtual routers.

10. Click **OK**.

    The Backbone/Service pane displays a list of services.

    For more information about viewing service information for action congestion points, see *Viewing Information about Action Congestion Points in a Backbone Network by Service* on page 344.

Viewing Information About Congestion Points in a Backbone Network with the C-Web Interface

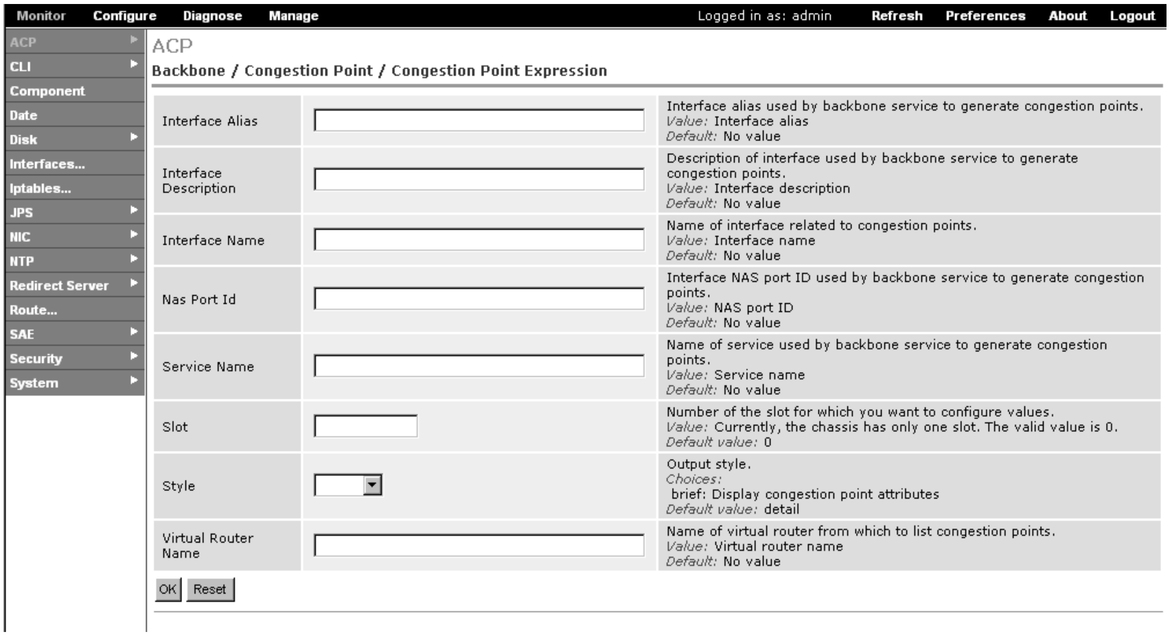
You can display information about congestion points in a backbone network by service or by DN.

Viewing Information About Congestion Points in a Backbone Network by Expression

To view information about congestion points by expression:

1. Click **ACP > Backbone > Congestion Point > Congestion Point Expression**.

   The Backbone/Congestion Point/Congestion Point Expression pane appears.



Interface Alias	Interface alias used by backbone service to generate congestion points. *Value:* Interface alias *Default:* No value
Interface Description	Description of interface used by backbone service to generate congestion points. *Value:* Interface description *Default:* No value
Interface Name	Name of interface related to congestion points. *Value:* Interface name *Default:* No value
Nas Port Id	Interface NAS port ID used by backbone service to generate congestion points. *Value:* NAS port ID *Default:* No value
Service Name	Name of service used by backbone service to generate congestion points. *Value:* Service name *Default:* No value
Slot	Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style	Output style. *Choices:* brief: Display congestion point attributes *Default value:* detail
Virtual Router Name	Name of virtual router from which to list congestion points. *Value:* Virtual router name *Default:* No value

2. In the Interface Alias box, enter the interface alias used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.

3. In the Interface Description box, enter the interface description used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.

4. In the Interface Name box, enter the name of an interface to display information about one interface, or leave the box empty to display information about all interfaces.

5. In the NAS Port ID box, enter the NAS port ID used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.

6. In the Service Name box, enter the name of a service to display information about one service, or leave the box empty to display information about all services.

7. In the Slot box, enter the number of the slot for which you want to display congestion point information.

8. Select an output style from the Style list.

9. In the Virtual Router Name box, enter a virtual router name to display information about a specific virtual router, or leave the box empty to display information about all virtual routers.

10. Click **OK**.

    The Backbone/Congestion Point/Congestion Point Expression pane displays a list of congestion points.

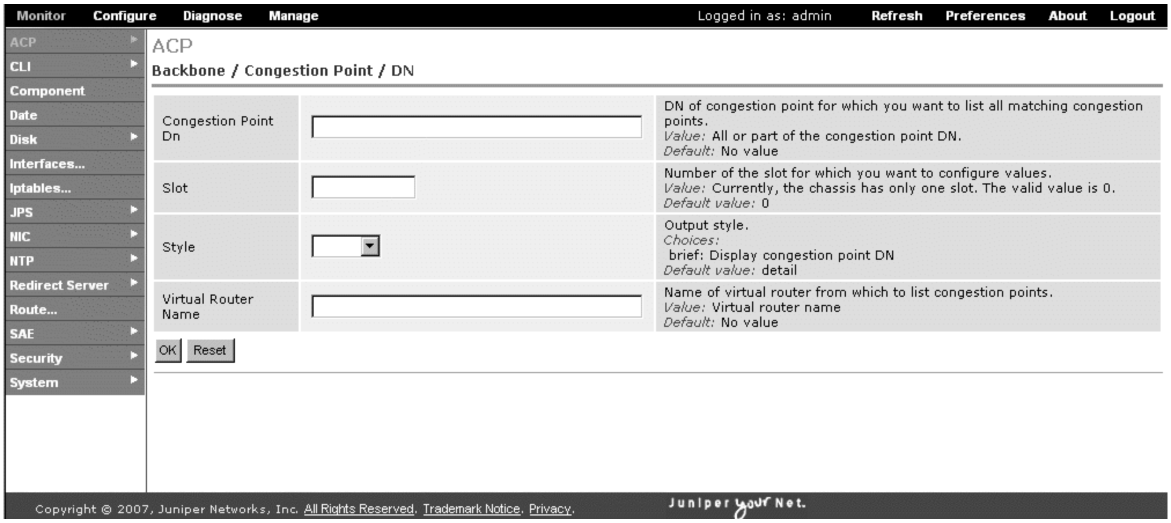
    For more information about viewing information for action congestion points by expression, see *Viewing Information about Action Congestion Points in a Backbone Network by Expression* on page 346.

Viewing Information About Congestion Points in a Backbone Network by DN

To view information about congestion points by DN:

1. Click **ACP > Backbone > Congestion Point > DN**.

   The Backbone/Congestion Point/DN pane appears.



ACP

Backbone / Congestion Point / DN

Congestion Point Dn	DN of congestion point for which you want to list all matching congestion points. *Value:* All or part of the congestion point DN. *Default:* No value
Slot	Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style	Output style. *Choices:* brief: Display congestion point DN *Default value:* detail
Virtual Router Name	Name of virtual router from which to list congestion points. *Value:* Virtual router name *Default:* No value

2. In the Congestion Point DN box, enter a full or partial congestion point name to display information about one or more specific congestion points, or leave the box empty to display information about all congestion points.

3. In the Slot box, enter the number of the slot for which you want to display congestion point information.

4. Select an output style from the Style list.

5. In the Virtual Router Name box, enter a virtual router name to display information about a specific virtual router, or leave the box empty to display information about all virtual routers.

6. Click **OK**.

   The Backbone/Congestion Point/DN pane displays a list of congestion points.

Viewing Information About Action Congestion Points in a Backbone Network with the C-Web Interface

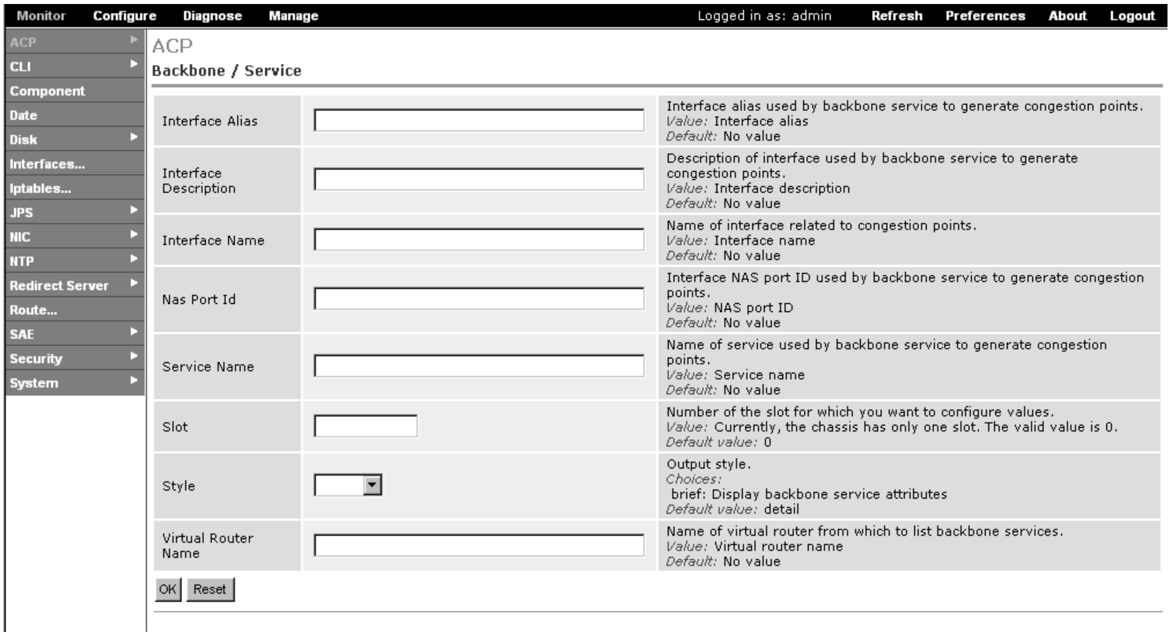
Backbone congestion points that are configured to run applications are action congestion points. You can view information about action congestion points by displaying congestion points in a backbone network by service or by expression.

Viewing Information about Action Congestion Points in a Backbone Network by Service

To view information about action congestion points in a backbone network by service:

1. Click **ACP > Backbone > Service**.

   The Backbone/Service pane appears.



ACP

Backbone / Service

Interface Alias	Interface alias used by backbone service to generate congestion points. *Value:* Interface alias *Default:* No value
Interface Description	Description of interface used by backbone service to generate congestion points. *Value:* Interface description *Default:* No value
Interface Name	Name of interface related to congestion points. *Value:* Interface name *Default:* No value
Nas Port Id	Interface NAS port ID used by backbone service to generate congestion points. *Value:* NAS port ID *Default:* No value
Service Name	Name of service used by backbone service to generate congestion points. *Value:* Service name *Default:* No value
Slot	Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style	Output style. *Choices:* brief: Display backbone service attributes *Default value:* detail
Virtual Router Name	Name of virtual router from which to list backbone services. *Value:* Virtual router name *Default:* No value

2. In the Interface Alias box, enter the interface alias used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.

3. In the Interface Description box, enter the interface description used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.

4. In the Interface Name box, enter the name of an interface to display information about one interface related to congestion points, or leave the box empty to display information about all interfaces.

5. In the NAS Port ID box, enter the NAS port ID used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.

6. In the Service Name box, enter the name of a service to display information about one service, or leave the box empty to display information about all services.

7. In the Slot box, enter the number of the slot for which you want to display congestion point information.

8. Select an output style from the Style list.

9. In the Virtual Router Name box, enter a virtual router name to display information about a specific virtual router, or leave the box empty to display information about all virtual routers.

10. Click **OK**.

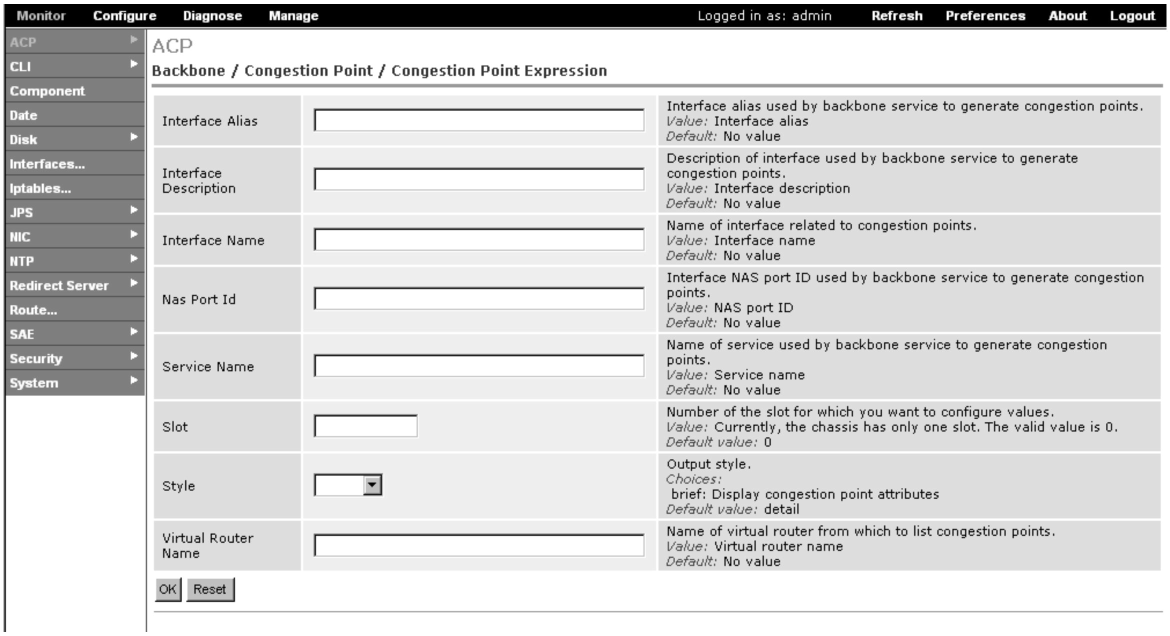
    The Backbone/Service pane displays a list of congestion points.

Viewing Information about Action Congestion Points in a Backbone Network by Expression

To view information about action congestion points in a backbone network by expression:

1. Click **ACP > Backbone > Congestion Point > Congestion Point Expression**.

   The Backbone/Congestion Point/Congestion Point Expression pane appears.



ACP

Backbone / Congestion Point / Congestion Point Expression

Interface Alias		Interface alias used by backbone service to generate congestion points. *Value:* Interface alias *Default:* No value
Interface Description		Description of interface used by backbone service to generate congestion points. *Value:* Interface description *Default:* No value
Interface Name		Name of interface related to congestion points. *Value:* Interface name *Default:* No value
Nas Port Id		Interface NAS port ID used by backbone service to generate congestion points. *Value:* NAS port ID *Default:* No value
Service Name		Name of service used by backbone service to generate congestion points. *Value:* Service name *Default:* No value
Slot		Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style		Output style. *Choices:* brief: Display congestion point attributes *Default value:* detail
Virtual Router Name		Name of virtual router from which to list congestion points. *Value:* Virtual router name *Default:* No value

2. In the Interface Alias box, enter the interface alias used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.

3. In the Interface Description box, enter the interface description used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.

4. In the Interface Name box, enter the name of an interface to display information about one interface related to congestion points, or leave the box empty to display information about all interfaces.

5. In the NAS Port ID box, enter the NAS port ID used by the backbone service to generate congestion points, or leave the box empty to display information about all interfaces.
6. In the Service Name box, enter the name of a service to display information about one service, or leave the box empty to display information about all services.
7. In the Slot box, enter the number of the slot for which you want to display congestion point information.
8. Select an output style from the Style list.
9. In the Virtual Router Name box, enter a virtual router name to display information about a specific virtual router, or leave the box empty to display information about all virtual routers.
10. Click **OK**.

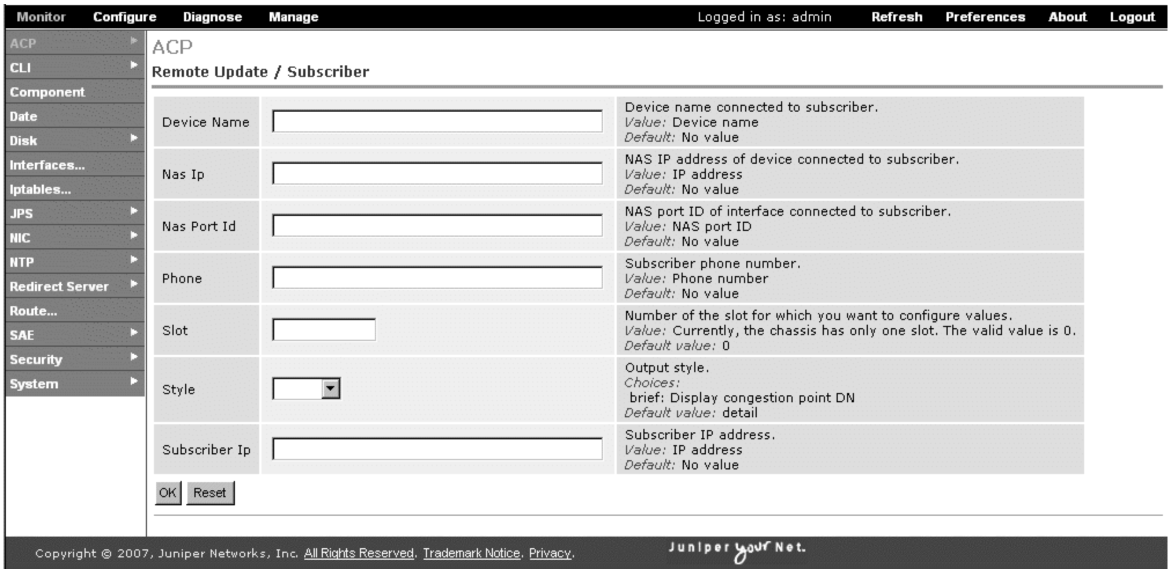
    The Backbone/Congestion Point/Congestion Point Expression pane displays a list of congestion points.

Viewing Information About Subscribers Obtained from External Applications with the C-Web Interface

To view information about subscribers obtained from external applications:

1. Click **ACP > Remote Update > Subscriber**.

    The Remote Update/Subscriber pane appears.



ACP

Remote Update / Subscriber

Device Name		Device name connected to subscriber. *Value:* Device name *Default:* No value
Nas Ip		NAS IP address of device connected to subscriber. *Value:* IP address *Default:* No value
Nas Port Id		NAS port ID of interface connected to subscriber. *Value:* NAS port ID *Default:* No value
Phone		Subscriber phone number. *Value:* Phone number *Default:* No value
Slot		Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style		Output style. *Choices:* brief: Display congestion point DN *Default value:* detail
Subscriber Ip		Subscriber IP address. *Value:* IP address *Default:* No value

2. In the Device Name box, enter the device name of the congestion point, or leave the box blank to display information about all devices.

3. In the NAS IP box, enter the NAS IP address of the device connected to the subscriber, or leave the box empty to display information about all subscribers.
4. In the NAS Port ID box, enter the NAS port ID connected to the subscriber, or leave the box empty to display information about all subscribers.
5. In the Phone box, enter the phone number of the subscriber, or leave the box blank to display information about all subscribers.
6. In the Slot box, enter the number of the slot for which you want to display external subscriber information.
7. Select an output style from the Style list.
8. In the Subscriber IP box, enter the subscriber IP address, or leave the box empty to display information about all subscribers.
9. Click **OK**.

   The Remote Update/Subscriber pane displays the congestion points.

Viewing Information About Congestion Points Added Through an External Application with the C-Web Interface

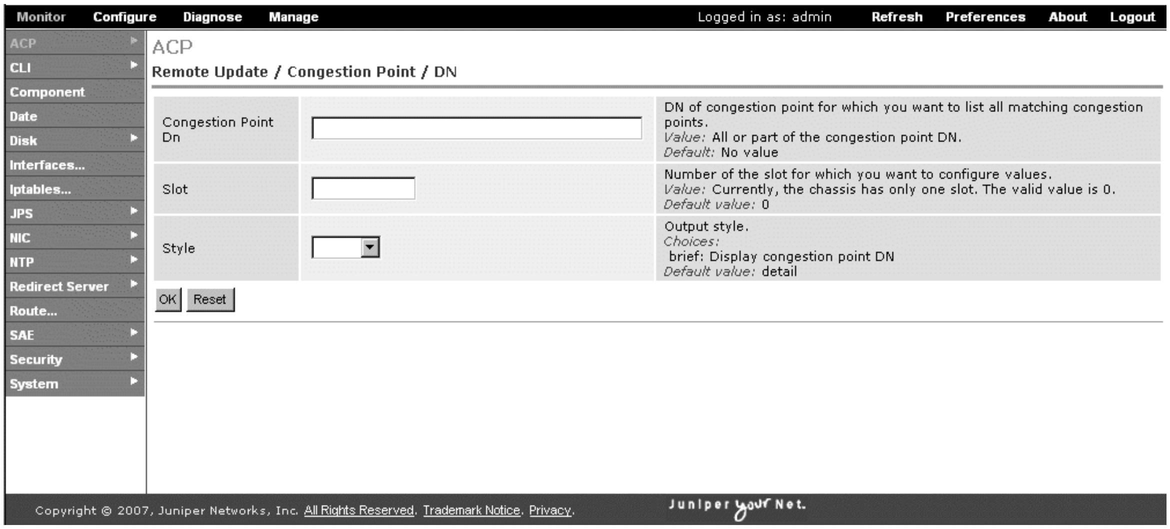
You can view information about congestion points added through an external application by DN or by interface name.

Viewing Information About Congestion Points from an External Application by DN

To view information about congestion points added through an external application by DN:

1. Click **ACP** > **Remote Update** > **Congestion Point** > **DN**.

   The Remote Update/Congestion Point/DN pane appears.



ACP

Remote Update / Congestion Point / DN

Congestion Point Dn	DN of congestion point for which you want to list all matching congestion points. *Value:* All or part of the congestion point DN. *Default:* No value
Slot	Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style	Output style. *Choices:* brief: Display congestion point DN *Default value:* detail

2. In the Congestion Point DN box, enter the DN of the congestion point, or leave the box blank to display information about all devices.

3. In the Slot box, enter the number of the slot for which you want to display congestion point information.

4. Select an output style from the Style list.

5. Click **OK**.

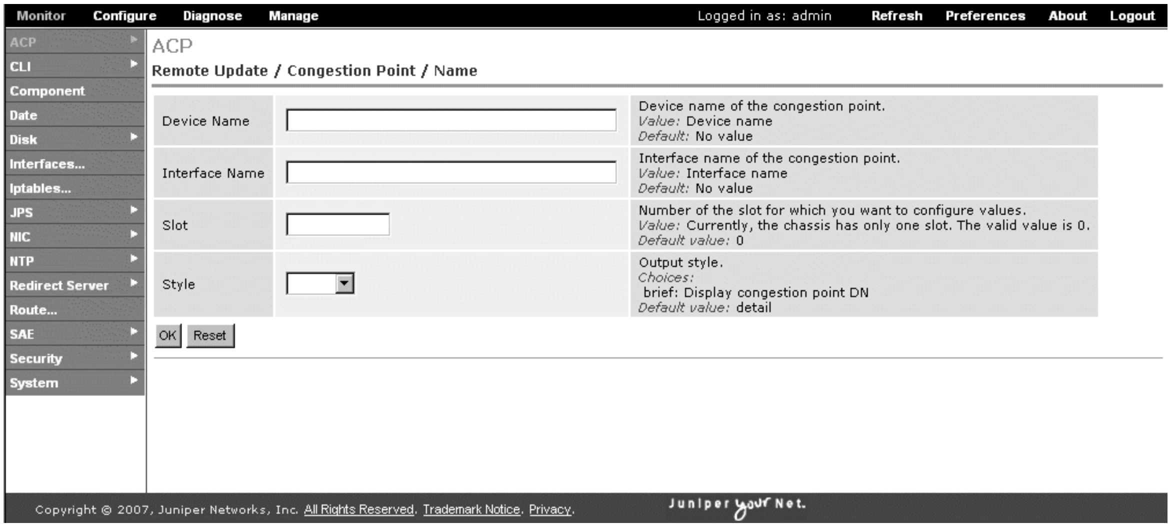
   The Remote Update/Congestion Point/DN pane displays the congestion points.

Viewing Information About Congestion Points from an External Application by Interface Name

To view information about congestion points added through an external application by interface name:

1. Click **ACP > Remote Update > Congestion Point > Name**.

   The Remote Update/Congestion Point/Name pane appears.



ACP

Remote Update / Congestion Point / Name

Device Name		Device name of the congestion point. *Value:* Device name *Default:* No value
Interface Name		Interface name of the congestion point. *Value:* Interface name *Default:* No value
Slot		Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style		Output style. *Choices:* brief: Display congestion point DN *Default value:* detail

2. In the Device Name box, enter the device name of the congestion point, or leave the box blank to display information about all devices.
3. In the Interface Name box, enter the interface name of the congestion point, or leave the box blank to display information about all interfaces.
4. In the Slot box, enter the number of the slot for which you want to display congestion point information.
5. Select an output style from the Style list.
6. Click **OK**.

   The Remote Update/Congestion Point/Name pane displays the congestion points.

Viewing Statistics for the SRC-ACP Configuration with the C-Web Interface

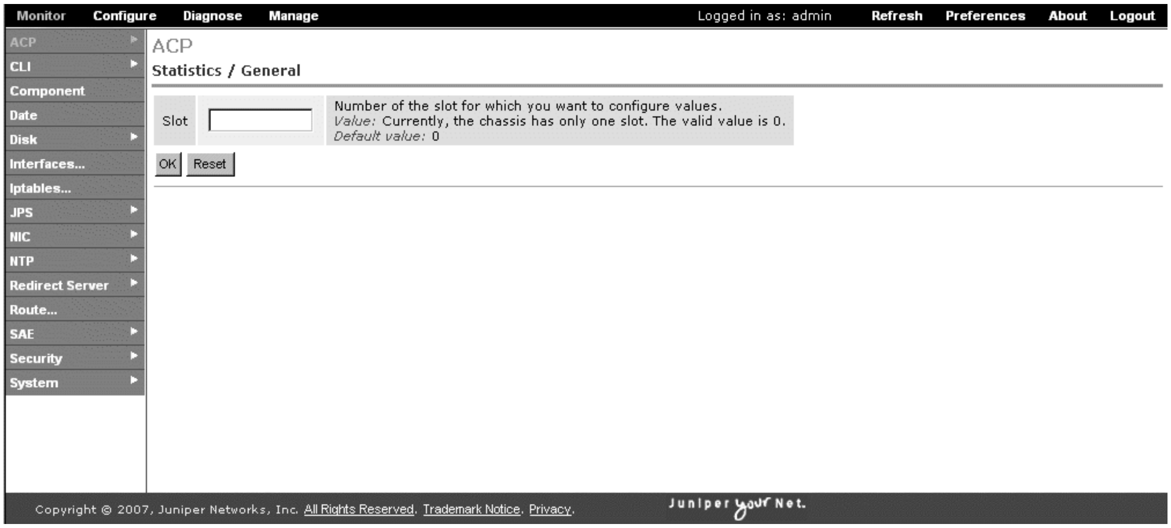
You can view general statistics for the SRC-ACP configuration. You can also view specific statistics for the directory and for virtual routers.

Viewing General Statistics for SRC-ACP

To view general statistics for SRC-ACP:

- Click **ACP > Statistics > General**.

  The Statistics/General pane appears.



ACP

Statistics / General

Slot

Number of the slot for which you want to configure values.
Value: Currently, the chassis has only one slot. The valid value is 0.
Default value: 0

7. In the Slot box, enter the number of the slot for which you want to display general statistics.
8. Click **OK**.

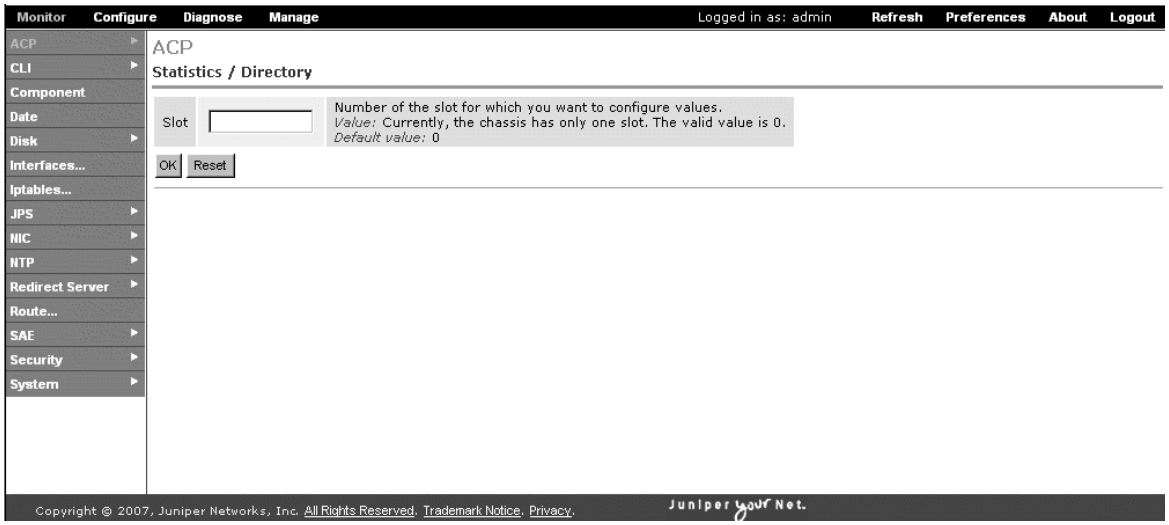
   The Statistics/General pane displays general SRC-ACP statistics.

Viewing Statistics for the SRC-ACP Directory

To view statistics about the SRC-ACP directory:

- Click **ACP > Statistics > Directory**.

  The Statistics/Directory pane appears.



ACP

Statistics / Directory

Slot

Number of the slot for which you want to configure values.
Value: Currently, the chassis has only one slot. The valid value is 0.
Default value: 0

9. In the Slot box, enter the number of the slot for which you want to display directory statistics.
10. Click **OK**.

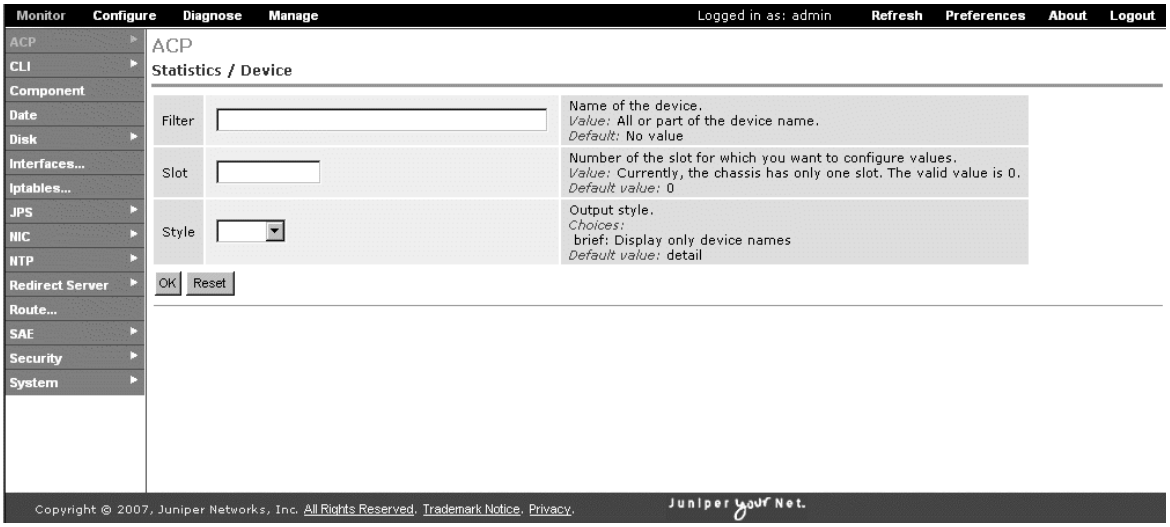
    The Statistics/Directory pane displays statistics for the SRC-ACP directory.

Viewing Device Statistics for SRC-ACP

To view device statistics for SRC-ACP:

1. Click **ACP > Statistics > Device**.

   The Statistics/Device pane appears.



ACP

Statistics / Device

Filter		Name of the device. *Value:* All or part of the device name. *Default:* No value
Slot		Number of the slot for which you want to configure values. *Value:* Currently, the chassis has only one slot. The valid value is 0. *Default value:* 0
Style		Output style. *Choices:* brief: Display only device names *Default value:* detail

2. In the Filter box, enter a substring of the virtual router name, or leave the box blank to display information for all virtual routers.
3. In the Slot box, enter the number of the slot for which you want to display device statistics.
4. Select an output style from the Style list.
5. Click **OK**.

   The Statistics/Device pane displays router statistics for SRC-ACP.

Chapter 26

Providing Admission Control with SRC-ACP on a Solaris Platform

This chapter describes how to install, configure, and manage the SRC Admission Control Plug-In (SRC-ACP) application 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 Controllers to configure SRC-ACP. See *Chapter 21, Configuring Admission Control with the SRC CLI*.

Topics in this chapter include:

- Installing SRC-ACP on page 354
- Configuring SRC-ACP on page 358
- Starting SRC-ACP on page 394
- Stopping SRC-ACP on page 394
- Monitoring and Managing SRC-ACP on page 394

For information about SRC-ACP, see *Chapter 20, Overview of Providing Admission Control with SRC-ACP*.

Installing SRC-ACP

SRC-ACP uses the SAE plug-in interface. You can install SRC-ACP on the same host as the SAE; doing so will simplify the SRC configuration. However, you may improve the performance of the SRC network by installing SRC-ACP on a separate host from the SAE.

The SRC-ACP package for Solaris platforms is located on the application library CD.

To install the SRC-ACP package:

1. On the UNIX host where you will install SRC-ACP, log in as `root`.
2. Place the application library CD in the CD drive.
3. Launch the **pkgadd** tool.

   ```
   pkgadd -d /cdrom/cdrom0/ACP_for_SDX/UMCacp
   ```

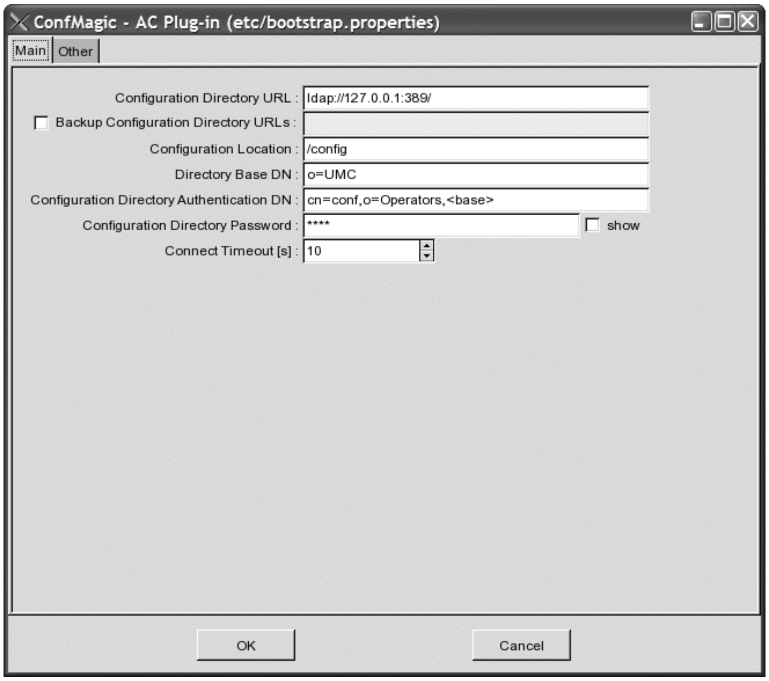
   The tool displays the license agreement.
4. Press Enter to move through the agreement, and then enter **y** to accept the license agreement when prompted by the tool.
5. Follow the prompt directions to accept the installation directory for the package, to permit the use of superuser scripts required for the package, and so on.

After you have installed the SRC-ACP application on a host, you must add the SRC-ACP configuration to the directory. To do so:

1. On the SRC-ACP host, log in as `root` or as another authorized administrator.
2. Launch the local configuration tool from the ACP installation directory.

   ```
   ./opt/UMC/acp/etc/config -l&
   ```

   The local configuration tool window appears.



Configuration Directory URL : ldap://127.0.0.1:389/
☐ Backup Configuration Directory URLs :
Configuration Location : /config
Directory Base DN : o=UMC
Configuration Directory Authentication DN : cn=conf,o=Operators,<base>
Configuration Directory Password : **** ☐ show
Connect Timeout [s] : 10

3. Complete the fields in the local configuration tool window. *Local Properties for SRC-ACP* on page 355 describes the fields on the Main tab and the Other tab.

4. Click **OK**.

5. A file called *bootstrap,properties* appears in the */opt/UMC/acp/etc* folder, and the SRC-ACP configuration appears in the directory. The SRC software automatically sets some values for properties that do not appear in the local configuration tool window.

Local Properties for SRC-ACP

Use the fields in this section to configure the local properties for SRC-ACP.

Configuration Directory URL

- URL of the primary directory.
- Value—URL in the format ldap://<host>:389
  - <host>—IP address or name of directory host
- Example—ldap://192.0.2.1:389/
- Property name—Config.java.naming.provider.url

Backup Configuration Directory URLs

- List of redundant directories.
- Value—Space-separated list of URLs; URLs have the format ldap://<host>:389
  - <host>—IP address or name of directory host
- Default—Unspecified
- Example—ldap://192.0.2.1:389/ ldap://192.0.2.3:389/
- Property name—Config.net.juniper.smgt.des.backup_provider_urls

Configuration Location

- Name of the object that contains the SRC-ACP configuration data.
- Value—/<objectName>
- Guidelines—This object appears in *l = ACP, ou = staticConfiguration, o = Management, o = umc*. Sensible choices for this field are the name of the host or the name of a given location. If you configure SRC-ACP redundancy, use the same name in the ACP configurations on both hosts.
- Default—/config
- Example—bostonConfig
- Property name—Config.ACP.namespace

Directory Base DN

- DN of the root of the SRC data in the directory.
- Value—*o =* <DN>
  - <DN>—DN
- Guidelines—If you are storing non-SRC data in the directory, and that data changes frequently whereas the SRC data does not, you may need to adjust the default value to improve performance. For optimal performance, set the value to the DN of an entry superior to both the SRC data and the changing non-SRC data.
- Default—*o = umc*
- Example—*o = umc*
- Property name—Config.net.juniper.smgt.des.event_baseDN

Configuration Directory Authentication DN

- DN of the directory entry that defines the username with which the SRC component accesses the directory.
- Value—<DN>
- Example—*cn = nic, ou = Components, o = Operators, <base>*
- Default—*cn = conf, o = Operators, <base>*
- Property name—Config.java.naming.security.principal

Configuration Directory Password

- Password with which SRC-ACP accesses the directory.
- 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
- Example—secret
- Property name—Config.java.naming.security.credentials

Connect Timeouts (s)

- Maximum time that the directory eventing system (DES) waits for the directory to respond.
- Value—Number of seconds in the range 1–2147483647
- Default—10
- Example—5
- Property name—Config.net.juniper.smgt.des.connect.timeout

Java Runtime Environment

- Path to the Java Runtime Environment (JRE).
- Value—Text string
- Guidelines—The SRC software requires a JRE that conforms to the Java 2 specification. The SRC software has been tested with Sun's JRE. Check the *SRC Release Notes* for information about which version of the Sun JRE is distributed with the SRC software. Other JREs should work but have not been tested with the SRC software.
- Default—*../jre/bin/java* (path for the JRE that is distributed with the SRC software and installed with the other SRC components)
- Property name—Acp.java

Java Heap Size

- Maximum amount of memory available to the JRE.
- Value—Number of megabytes in the format <integer>m
- Guidelines—Change this value if you have problems caused by lack of memory. Set the value lower than the available physical memory to avoid low performance caused by disk swapping.
- Default—64m

Configuring SRC-ACP

To use SRC-ACP in the SRC network, you must perform some configuration. For information about these configuration procedures, see:

1. Configuring the SAE for SRC-ACP on page 358
2. Configuring SRC-ACP Properties on page 361
3. (Edge and dual mode only) Configuring SRC-ACP to Manage the Edge Network on page 376
4. (Backbone and dual mode only) Configuring SRC-ACP to Manage the Backbone Network on page 382

Configuring the SAE for SRC-ACP

You must configure the SAE to recognize SRC-ACP by adding some information about SRC-ACP to the SAE properties. To do so:

1. Configure SRC-ACP as an external plug-in for the SAE.
2. Configure event publishers.
3. (Backbone and dual mode only) Optionally, configure a hosted plug-in that monitors the state of interfaces on VRs.

Configuring SRC-ACP as an External Plug-In

To configure an external plug-in for the SAE, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 10, Overview of Configuring Plug-Ins for Solaris Platforms*. Use the following values for the plug-in attributes:

- Value for edge and dual modes—PA_UPSTREAM_BANDWIDTH, PA_DOWNSTREAM_BANDWIDTH, PA_SERVICE_NAME, PA_ROUTER_NAME, PA_LOGIN_NAME, PA_USER_DN, PA_PORT_ID, PA_SESSION_ID, PA_USER_IP_ADDRESS, PA_NAS_IP, PA_USER_SESSION_ID, PA_EVENT_TIME
- Value for backbone mode—PA_UPSTREAM_BANDWIDTH, PA_DOWNSTREAM_BANDWIDTH, PA_SERVICE_NAME, PA_ROUTER_NAME, PA_SESSION_ID, PA_NAS_IP, PA_EVENT_TIME

Configuring Event Publishers

You must configure the SAE to publish the following types of events to SRC-ACP:

- (Edge and dual mode only) Global user tracking
- Global service authorization
- Global service tracking

For information about configuring event publishers, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 11, Configuring Accounting and Authentication Plug-Ins with the SRC CLI*. Identify the instance of SRC-ACP by the name of the host on which you configured it.

Configuring the SAE to Monitor Interfaces for Congestion Points

NOTE: Configure this feature only if SRC-ACP is in backbone or dual mode.

The SAE uses a hosted internal plug-in to monitor the state of interfaces on a VR for backbone congestion points. If a subscriber tries to activate a service on an interface that is unavailable, the SAE denies the request. The plug-in also monitors the directory for new backbone congestion points. To configure an internal plug-in for the SAE, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 10, Overview of Configuring Plug-Ins for Solaris Platforms* and use the properties described in *Internal Plug-In Properties for Monitoring Congestion Points* on page 359.

When this plug-in initializes, it reads all the backbone services from the directory and generates a list of the DNs (network interfaces) of the backbone congestion points. The SAE sends interface tracking events, which contain the names of the interfaces, VRs, and routers to this plug-in. For this feature to work correctly, the interface, VR, and router must appear in *o = AdmissionControl, o = umc* (see *Configuring Network Interfaces in the Directory* on page 382).

Internal Plug-In Properties for Monitoring Congestion Points

Use the descriptions in this section to configure an internal plug-in for the SAE.

Plug-in Class

- Class name of the plug-in.
- Value—net.juniper.smgt.sae.plugin.ACPIntfListener

Plugin.acpIntfListener.host

- IP address or name of the host that supports the directory that contains backbone service definitions and network interfaces.
- Value—Plugin.acpIntfListener.host = <host>:<port>
  - <host>—IP address or name of host
  - <portNumber>—Number of the TCP port; default is 389
- Default—Plugin.acpIntfListener.host = 127.0.0.1

Plugin.acpIntfListener.bindDN

- DN of the directory entry that defines the username with which the plug-in accesses the directory.
- Value—DN
- Default—Plugin.acpIntfListener.bindDN = cn=umcadmin, <base>

Plugin.acpIntfListener.bindPW

- Password with which the plug-in accesses the directory.
- Value—Text string
- Default—Plugin.acpIntfListener.bindPW = ssp
- Example—Plugin.acpIntfListener.bindPW = secret

Plugin.acpIntfListener.baseDN

- DN at which SRC-ACP stores backbone congestion points.
- Value—DN
- Default—Plugin.acpIntfListener.baseDN = *o=CongestionPoints, <base>*

Plugin.acpIntfListener.acpBaseDN

- DN at which SRC-ACP stores edge congestion points.
- Value—DN
- Default—Plugin.acpIntfListener.acpBaseDN = *o=AdmissionControl, <base>*

Plugin.acpIntfListener.timeout

- Maximum time that the plug-in waits for the router to respond.
- Value—Number of milliseconds in the range 0–2147483647
  - 0—No timeout
  - Other values—Actual time
- Default—Plugin.acpIntfListener.timeout = 5000

Plugin.acpIntfListener.objectref = objectref

- Object reference for the ACP plug-in, as defined in the field ACP.ior in the ACP's CORBA properties (see *Configuring SRC-ACP Properties* on page 361).

Plugin.acpIntfListener.<standardJNDISuffix>
Plugin.acpIntfListener.des.net.juniper.smgt.des.<property suffix>

- Standard Java Naming and Directory Interface (JNDI) and DES properties. For complete information about these properties, see *SRC-PE Getting Started Guide, Chapter 37, Distributing Directory Changes to SRC Components on a Solaris Platform*. The following list shows the properties you should include, with suggested values.
    - Plugin.acpIntfListener.securityProtocol = ldaps
    - Plugin.acpIntfListener.des.net.juniper.smgt.des.event_baseDN = o=CongestionPoints, <base>
    - Plugin.acpIntfListener.des.net.juniper.smgt.des.enable_sysman = true
    - Plugin.acpIntfListener.des.net.juniper.smgt.des.polling_interval = 30
    - Plugin.acpIntfListener.des.net.juniper.smgt.des.connect.timeout = 10
    - Plugin.acpIntfListener.des.net.juniper.smgt.des.retry_interval = 60
    - Plugin.acpIntfListener.des.net.juniper.smgt.des.connectioncheck_interval = 60
    - Plugin.acpIntfListener.des.net.juniper.smgt.des.signatureDN = <base>
    - Plugin.acpIntfListener.des.net.juniper.smgt.lib.config.polling_timeout = 10
    - Plugin.acpIntfListener.des.net.juniper.smgt.des.enable_eventing = true
    - Plugin.acpIntfListener.des.net.juniper.smgt.des.share_connection = false
    - Plugin.acpIntfListener.des.net.juniper.smgt.des.connection_manager_id = ACPIntfListener

Configuring SRC-ACP Properties

To configure SRC-ACP properties, you can use SDX Admin to modify ACP properties in *l=ACP, ou=staticConfiguration, ou=Configuration, o=Management, o=umc*.

You can also use the SRC CLI. See *Chapter 21, Configuring Admission Control with the SRC CLI*.

Configuring Logging

To configure logging, see *Configuring Logging Destinations for SRC-ACP* on page 294. For information about configuring logging, see *SRC-PE Monitoring and Troubleshooting Guide, Chapter 5, Managing SRC Log Files on a Solaris Platform*.

Configuring SRC-ACP Operation

To configure how SRC-ACP operates:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry *l = ACP, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*.
3. Click the Main tab in the ACP Configuration pane.
4. Edit or accept the default values for the properties that configure how SRC-ACP operates. See *SRC-ACP Operating Properties* on page 362.

SRC-ACP Operating Properties

In SDX Admin, you can modify the following SRC-ACP operating properties in the content pane for an SRC-ACP configuration.

ACP.backup.dir

- Folder that stores backup information about subscribers, services, and congestion points.
- Value—Absolute path or a path relative to *opt/UMC/acp*
- Default—backup

ACP.mode

- Regions of the network that SRC-ACP manages.
- Value
  - Edge—SRC-ACP operates in the edge region of network only.
  - Backbone—SRC-ACP operates in the backbone region of the network only.
  - Dual—SRC-ACP operates in both the edge and backbone regions of network.
- Default—Dual

ACP.backup.cacheSize

- Number of plug-in events from the SAE that SRC-ACP can store in its cache. Specifying a large number increases the efficiency of SRC-ACP, and minimizes the use of CPU resources; however, the amount of memory available for the cache will depend on the host’s resources.
- Value—Integer in the range 0–4294967295
- Default—10000

ACP.overloadControl

- Specifies how SRC-ACP deals with situations where the components exceed the allocated bandwidth because the service was activated after the authorization was granted.
- Value—Integer in the range –1–4294967295
  - –1—SRC-ACP ignores overload
  - Integer greater than or equal to 0 – bandwidth (in bps) by which the maximum may be exceeded
- Default—0

ACP.INTFAutoCompletion

- Specifies whether SRC-ACP uses the information acquired from the router to determine the congestion points (see *Deriving Edge Congestion Points* on page 275).
- Value
  - Yes—Information used
  - No—Information not used
- Default—No

ACP.BackgroundBandwidthTuningFactors

- Specifies factors that compensate for actual use of bandwidth, as opposed to allocated bandwidth.
- Value—List of tuning factors, separated by commas; each tuning factor is a floating number in the range 0–1
- Default—None
- Example—0.8, 0.9

UserBandwidthExceed.message

- Error message that SRC-ACP sends when the subscriber exceeds the allocated bandwidth.
- Value—Text string
- Default—User bandwidth exceeded

networkBandwidthExceed.message

- Error message that SRC-ACP sends when traffic flow exceeds the allocated bandwidth on an interface between the subscriber and the router.
- Value—Text string
- Default—Network bandwidth exceeded

ACP.backupDb.reorganizationSize

- Value by which the sum of the sizes of the files that contain ACP data can increment before SRC-ACP reorganizes the files. Reorganizing the files reduces their size.
- Value—Text string in the format <number>m or <number>g
  - <number>m—Size of database in megabytes
  - <number>g—Size of database in gigabytes
- Default—100m
- Example—1g
- Guidelines—Choose a value that is significantly lower than the capacity of the machine's hard disk.

ACP.indexedDB.keys

- Values to look for in the configuration data. Specifying index keys can improve performance by filtering the data.
- Value—List of attributes, separated by commas; an attribute is one of the following text strings
  - accountingId—Value of directory attribute accountingUserId.
  - dhcpPacket—Content of the DHCP discover request.
  - hostname— Name of the host on which the SAE is installed.
  - ifIndex—SNMP index of the interface. This attribute is not supported on JUNOS routing platforms.
  - ifRadiusClass—RADIUS class attribute on the JUNOSe interface. This attribute is not supported on JUNOS routing platforms.
  - ifSessionId—Identifier for RADIUS accounting on the JUNOSe interface. This attribute is not supported on JUNOS routing platforms.
  - interfaceAlias—Alias of the interface; that is, the IP description in the interface configuration.
  - interfaceDescr—SNMP description of the interface.
  - interfaceName—Name of the interface.
  - loginName—Subscriber's login name.
  - nasInetAddress—IP address of the router; using a byte array instead of an integer.
  - nasPort—NAS port used by the router to identify the interface to RADIUS.
  - portId—Identifier of VLAN or virtual circuit. For a virtual circuit, use the format <VPI>/<VCI>. This attribute is not supported on JUNOS routing platforms.
    - ❑ <VPI>—Virtual path identifier
    - ❑ <VCI>—Virtual connection identifier
  - primaryUserName—PPP login name or the public DHCP username. This attribute is not supported on JUNOS routing platforms.

  - routerName—Name of the virtual router in the format <virtualRouter>@<router>.
    - <virtualRouter>—Virtual router name
    - <router>—Router name
  - routerType—Type of router driver.
  - userInetAddress—IP address of the subscriber; using a byte array instead of an integer.
  - userMacAddress—MAC address of the DHCP subscriber. This attribute is not supported on JUNOS routing platforms.
  - userRadiusClass—RADIUS class attribute of the subscriber session for a service. This attribute can occur multiple times and can be returned by an authorization plug-in.
  - userType—Type of subscriber.
- Default—interfaceName, routerName

ACP.interfaceTracking.filters

- Interface tracking event to be ignored by SRC-ACP. Filtering the interface tracking events can improve performance and can reduce the amount of memory required for keeping the congestion points updated.
- Value—Filter strings in the format of a list of <attribute> = <value> pairs; that can be contained within query operations
  - <attribute>—Name of an attribute for an interface tracking event. See value for the field ACP.indexedDB.keys on page 364.
  - <value>—Filtering string of the following types:
    - *—Any value
    - Explicit string—Any value matching the specified string (not case-sensitive)
    - String containing an asterisk—Any value containing the specified string (not case-sensitive)
  - To perform query operations on filter strings, you can use the following values in your filter strings:
    - ()—Match no objects.
    - (*)—Match all objects.
    - (&<filter> <filter>...)—Performs logical AND operation on filter strings; true if all filter strings match.
    - (|<filter> <filter>...)—Performs logical OR operation on filter strings; true if at least one filter string matches.
    - (!<filter>)—Performs logical NOT operation on filter string; true if the filter string does not match.
- Default—*
- Example—(& (interfaceName=fastEthernet3/0) (routerName=default@erx))

Configuring CORBA Interfaces

To configure CORBA interfaces for SRC-ACP:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry *l = ACP, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*.
3. Click the Main tab in the ACP Configuration pane.
4. Edit or accept the default values for the properties that configure CORBA interfaces for SRC-ACP. See *CORBA Interfaces Properties* on page 366.

CORBA Interfaces Properties

In SDX Admin, you can modify the following SRC-ACP properties in the content pane for an SRC-ACP configuration.

ACP.ior

- Exports the object reference for SRC-ACP through either a local file or a Common Object Services (COS) naming service.
- Values—One of the following references
  - file://<path>—Exports object reference through a local file
    - <path>—Absolute path to local file
  - corbaname::<cosNameServer>#<KEY>—Exports object reference through COS naming services
    - <cosNameServer>—IP address or Domain Name System (DNS) name of COS naming server
    - <KEY>—Object reference of SRC-ACP
  - corbaname:rir#<KEY>—Exports object reference through COS naming service; resolve-initial-references (rir) function finds DNS name of COS naming server
- Default—file:///var/acp/acp.ior

ACP.syncRateAdapter.ior

- Object reference for the ACP external interface.
- Value—See value for the field ACP.ior on page 366.
- Default—file:///var/acp/sra.ior
- Property name—ACP.syncRateAdapter.ior

Configuring SRC-ACP Redundancy

To configure SRC-ACP redundancy:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry *l = ACP, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*.
3. Click the Main tab in the ACP Configuration pane.
4. Edit or accept the default values for the properties that configure SRC-ACP redundancy. See *SRC-ACP Redundancy Properties* on page 367.

SRC-ACP Redundancy Properties

In SDX Admin, you can modify the following SRC-ACP properties in the content pane for an SRC-ACP configuration.

ACP.redundancy.enable

- Enables or disables SRC-ACP redundancy.
- Value
  - Yes—Enable redundancy.
  - No—Disable redundancy.
- Default—No

ACP.redundancy.local.ior

- In a redundant SRC-ACP configuration, exports the object reference for this SRC-ACP through a local file or COS naming service.
- Value—See value for the field ACP.ior on page 366.
- Default—None
- Example—corbaname::cosHost#0000000000000035...

ACP.redundancy.remote.ior

- In a redundant SRC-ACP configuration, exports the object reference for the other SRC-ACP through a local file or COS naming service.
- Value—See value for the field ACP.ior on page 366.
- Default—None
- Example—corbaname:rir#0000000000000035...

ACP.IgnoreUserOutOfSync

- Specifies whether user tracking events should be ignored when they raise an OutOfSync exception to the SAE when state synchronization is enabled. SRC-ACP raises an OutOfSync exception when SRC-ACP handles service tracking or authentication events without receiving a user start event first.
- Value
  - true—Ignore user tracking events that raise an OutOfSync exception.
  - false—Tracks all events; SRC-ACP raises an OutOfSync exception.
- Default—false

ACP.redundancy.bkpns.ior

- In a redundant SRC-ACP configuration, exports the object reference for the backup naming service through a local file or COS naming service. The primary SRC-ACP registers the IOR and redundancy IOR to both naming services, while the secondary SRC-ACP registers the redundancy IOR to both naming services.
- Value—See value for the field ACP.ior on page 366.
- Default—None
- Example—corbaname::cosHost#0000000000000035...

Configuring State Synchronization

Enabling state synchronization can affect performance because of resource consumption. To configure state synchronization with SAE:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry *l = ACP, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*.
3. Click the Main tab in the ACP Configuration pane.
4. Edit or accept the default values for the properties that configure state synchronization. See *State Synchronization Properties* on page 368.

State Synchronization Properties

In SDX Admin, you can modify the following SRC-ACP properties in the content pane for an SRC-ACP configuration.

ACP.stateSync

- Enables or disables SRC-ACP state synchronization with the SAE.
- Value
  - Yes—Enable state synchronization.
  - No—Disable state synchronization.
- Default—Yes

ACP.stateSyncBulkSize

- Number of events the SAE sends to SRC-ACP in a single method call during state synchronization.
- Value—Integer
- Default—1

ACP.redundancy.community.heartbeat

- Time interval for community members to check each other's availability when both redundancy and state synchronization are enabled.
- Value—Number of seconds in the range 0–4294967295
- Default—30

ACP.redundancy.community.acquire_timeout

- Time to wait before trying to reacquire the distributed lock when both redundancy and state synchronization are enabled.
- Value—Number of seconds in the range 0–4294967295
- Default—15

ACP.redundancy.community.blackout_time

- Time to wait before regaining control when both redundancy and state synchronization are enabled.
- Value—Number of seconds in the range 0–4294967295
- Default—30

Configuring Connections to the Subscribers' Directory

To configure how SRC-ACP connects to the directory that contains subscriber information:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry *l = ACP, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*.
3. Click the Main tab in the ACP Configuration pane.
4. Edit or accept the default values for the properties that configure connections to the subscriber's directory. See *Subscriber Directory Properties* on page 370.

Subscriber Directory Properties

In SDX Admin, you can modify the following SRC-ACP properties in the content pane for an SRC-ACP configuration.

NOTE: In the following descriptions, use the property name with the GlobalUserDatabase prefix if one directory supports all the components in the network. Use the property name with the <vrGroupName> prefix if you have partitioned the directory to provide information for different parts of the network through different VRs. For more information, see *Working with Partitioned Directories* on page 374.

---

server.address

- List of primary and redundant servers that manage data for subscribers.
- Value—List of IP addresses or hostnames separated by spaces
- Default—127.0.0.1
- Example—10.227.7.153

server.port

- TCP port for the directory.
- Value—Valid TCP port number
- Default—389

server.baseDN

- DN of the root of the directory.
- Value—List of attribute = value pairs separated by commas
- Default—*o=users, o=umc*

server.authDN

- DN used to authorize connections to the directory.
- Value—List of attribute = value pairs separated by commas
- Default—*cn=umcadmin, o=umc*

server.password

- Password used to authorize connections to the directory.
- Value—Text string
- Default—admin123

server.event_baseDN

- DN of the directory that contains event information.
- Value—DN
- Default—*o=umc*

des.enable_eventing

- Enables or disables directory eventing.
- Value
    - Yes—Enable directory eventing.
    - No—Disable directory eventing.
- Default—Yes

des.pollinginterval

- Time interval at which the SRC component polls the directory.
- Value—Number of seconds in the range 15–2147483647
- Default—30

server.intf_eventing

- Enables or disables directory eventing for congestion points.
- Value
    - Yes—Enable directory eventing.
    - No—Disable directory eventing.
- Default—No

Configuring Connections to the Services’ Directory

To configure how SRC-ACP connects to the directory that contains information about services:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry *l = ACP, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*.
3. Click the Main tab in the ACP Configuration pane.
4. Edit or accept the default values for the properties that configure connections to the service’s directory. See *Service Directory Properties* on page 372.

Service Directory Properties

In SDX Admin, you can modify the following SRC-ACP properties in the content pane for an SRC-ACP configuration.

NOTE: In the following descriptions, use the property name with the GlobalServiceDatabase prefix if one directory supports all the components in the network. Use the property name with the <vrGroupName> prefix if you have partitioned the directory to provide information for different parts of the network through different VRs. For more information, see *Working with Partitioned Directories* on page 374.

---

server.address

- List of primary and redundant servers that manage data for services.
- Value—List of IP addresses or hostnames separated by spaces
- Default—127.0.0.1
- Example—10.227.7.153 10.227.7.125

server.port

- TCP port for the directory.
- Value—Valid TCP port number
- Default—389

server.baseDN

- DN of the root of the directory that stores data about services.
- Value—List of attribute = value pairs separated by commas
- Default—*o=services, o=umc*

server.authDN

- DN that SRC-ACP uses to authorize connections to the directory that stores data about services.
- Value—List of attribute = value pairs separated by commas
- Example—*cn=umcadmin, o=umc*

server.password

- Password that SRC-ACP uses to authorize connections to the directory that stores data about services.
- Value—Text string
- Default—admin123

server.event_baseDN

- DN of the directory that contains event information for services.
- Value—DN
- Example—*o = umc*

des.enable_eventing

- Enables or disables directory eventing.
- Value
  - Yes—Enable directory eventing
  - No—Disable directory eventing
- Default—Yes

des.pollinginterval

- Time interval at which SRC-ACP polls the directory.
- Value—Number of seconds in the range 15–2147483647
- Default—30

server.intfBaseDN

- DN of the directory that contains information about network interfaces for edge congestion points.
- Value—DN
- Default—*o = AdmissionControl, o = umc*

server.congestionPointBaseDN

- DN of the directory that contains information about network interfaces for backbone congestion point objects.
- Value—DN
- Default—*o = CongestionPoints, o = umc*

eventing.reloadCongestionPoints

- Specifies whether SRC-ACP detects changes in the backbone congestion point for a service while SRC-ACP is operative.
- Value
  - Yes—SRC-ACP uses new information for the backbone congestion point as soon as SRC-ACP detects a change to the data in the directory.
  - No—SRC-ACP uses new information for the backbone congestion point only after you stop and restart SRC-ACP.
- Guidelines—Set this field to Yes only when you want to modify a congestion point (see on page 394). When you have modified the congestion point, reset this field to No.
- Default—No

server.intf_eventing

- Enables or disables directory eventing for congestion points.
- Value
  - Yes—Enable directory eventing.
  - No—Disable directory eventing.
- Default—No

Configuring Eventing Properties for Databases

You can configure all directory eventing properties with SDX Admin for the subscriber and service databases. For information about configuring directory eventing properties, see *SRC-PE Getting Started Guide, Chapter 37, Distributing Directory Changes to SRC Components on a Solaris Platform*. If one directory supports all the components in the network, use the following constructions for the properties:

- GlobalUserDatabase.des.<property>

- GlobalServiceDatabase.des.<property>

If, however, you have partitioned the directory, see *Working with Partitioned Directories* on page 374.

NOTE: For SRC-ACP, always set the value of the des.dispatcher_pool_size property to 1.

Working with Partitioned Directories

If you have partitioned the directory to provide information for different parts of the network through different VRs, you must define the Data Manager properties with SDX Admin. To do so:

1. Access SDX Admin.

2. In the navigation pane, highlight the entry *l=ACP, ou=staticConfiguration, ou=Configuration, o=Management, o=umc*.

3. Click the Main tab in the ACP Configuration pane.

4. Define the name of the VR group for each directory in the Property field by using the following formats.

   - VrUserDatabase.<vrGroupName>

   - VrServiceDatabase.<vrGroupName>

   <vrGroupName> is an arbitrary name that identifies the group of VRs in the partition.

5. Define the VRs in the group by defining the property <vrGroupName>.vrs.

6. Define the properties for connecting to the subscribers' directory and the services' directory by using the following format:

   <vrGroupName> <property> = <value>

   For information about these properties, see *Configuring Connections to the Subscribers' Directory* on page 369 and *Configuring Connections to the Services' Directory* on page 371.

7. Define directory eventing properties for the subscriber's directory and the services' directory by using the format

   <vrGroupName>.<des.Name>.des.<property> = <value>

   For more information, see *Configuring Eventing Properties for Databases* on page 374.

<vrGroupName>.vrs

- List of VRs that support the user database or the service database.
- Value—List of VRs separated by spaces in the format:
  <vrName>@<routerName>
  - <vrName>—Name of VR configured on the router
  - <routerName>—Name of router on which the VR is configured
- Example—vrdb1.vrs = default@erx1 default@erx2 vr1@erx3

Example In this example, the name of the VR group is vr1, and the group contains one VR called bigfoot@erx1.

```
VrUserDatabase.vr1
vr1.factory.class.name = net.juniper.smgt.acp.UserLdapDataManagerFactory
vr1.server.address = 127.0.0.1
vr1.server.port = 389
vr1.server.baseDN = o=users,o=umc
vr1.server.authDN = cn=umcadmin,o=umc
vr1.server.password = admin123
vr1.server.event_baseDN = o=umc
vr1.server.signatureDN = o=umc
vr1.des.enable_eventing = true
vr1.des.pollinginterval = 30
vr1.des.delegate_factory_initial = com.sun.jndi.ldap.LdapCtxFactory
vr1.des.connection_pool_size = 1
vr1.des.dispatcher_pool_size = 1
vr1.des.fake_delete = false
vr1.des.show_fake_delete = false
vr1.vrs = bigfoot@default
```

Configuring SRC-ACP Scripts and Classification

To configure scripts and classification, see *Configuring SRC-ACP Scripts and Classification* on page 305.

Configuring SRC-ACP to Manage the Edge Network

To configure SRC-ACP to manage the edge network you must:

1. Configure network interfaces that represent locations of congestion points in the directory.
2. Configure guaranteed bandwidths for subscribers.
3. Assign network interfaces to subscribers.
4. Configure guaranteed bandwidths for services.

See the following sections for details about these procedures.

You can configure objects in the directory by means of an LDAP client or by means of a network management database. These sections provide information about the LDAP attributes you must configure and their positions in the LDAP schema, as well as details on how to configure objects with SDX Admin. For detailed information about the LDAP schema, see the documentation on the SRC software distribution in the folder */SDK/doc/ldap*.

Configuring Network Interfaces in the Directory

You must add network interfaces to the directory. For the edge network, you do so by specifying in the DN *o = Admission Control, o = umc* the network interfaces of the routers and the switches in the access network between subscribers and the SRC network. Table 18 shows the object class for network interfaces and the associated attributes.

Table 18: SRC-ACP Information for Network Interfaces

Information	Mandatory or Optional Information	LDAP Schema
Network interface	Mandatory	networkInterface (object class)
Provisioned downstream bandwidth	Mandatory	downstreamProvisionedRate (attribute)
Provisioned upstream bandwidth	Mandatory	upstreamProvisionedRate (attribute)
List of downstream background bandwidths Entries separated by commas in the LDAP schema	Optional (For information about background bandwidths, see *Allocating Bandwidth to Applications Not Controlled by SRC-ACP* on page 277.)	downstreamBackgroundRate (attribute)
List of upstream background bandwidths Entries separated by commas in the LDAP schema	Optional (For information about background bandwidths, see *Allocating Bandwidth to Applications Not Controlled by SRC-ACP* on page 277.)	upstreamBackgroundRate (attribute)

To configure a network interface with SDX Admin:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry *o = Admission Control, o = umc*, and right-click.

3. Select **New > NetworkDevice**.

   The New Network Device dialog box appears.

4. Enter the name of the network device, and click **OK**.

   An object for the new network device appears in the navigation pane, and basic details for the new network device appear in the Main tab of the NetworkDevice pane.

5. (Optional) Enter a description for the network device in the Description field, and click Save in the NetworkDevice pane.

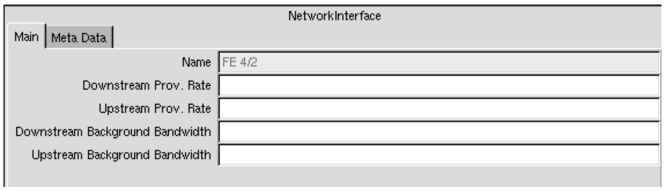
6. In the navigation pane, highlight the network device, and right-click.

7. Select **New > Network Interface**.

   The New Network Interface dialog box appears.

8. Enter the name of the network interface, and click **OK**.

   An object for the new network interface appears in the navigation pane, and basic details for the new network interface appear in the Main tab of the NetworkInterface pane.



Name: FE 4/2

9. Complete the fields using the information in *Bandwidth for Network Interfaces* on page 377.

10. Click **Save** in the NetworkInterface pane.

Bandwidth for Network Interfaces

Use the fields in this section to define bandwidth for network interfaces.

Downstream Prov. Rate

- Provisioned downstream bandwidth.
- Value—Number of bits per second

Upstream Prov. Rate

- Provisioned upstream bandwidth.
- Value—Number of bits per second

Downstream Background Bandwidth

- Downstream background bandwidths.
- Value—List of bandwidths separated by commas.
- Guidelines—Optional. For information about background bandwidths, see *Allocating Bandwidth to Applications Not Controlled by SRC-ACP* on page 277.

Upstream Background Bandwidth

- Upstream background bandwidths.
- Value—List of bandwidths separated by commas.
- Guidelines—Optional. For information about background bandwidths, see *Allocating Bandwidth to Applications Not Controlled by SRC-ACP* on page 277.

Configuring Bandwidths for Subscribers

You must configure bandwidths for subscribers that SRC-ACP manages in the edge region of the network.

If the access network between the subscriber and the router uses ATM, and all the traffic coming from one DSLAM travels on a single virtual path, you do not need to provision bandwidths for each subscriber. In this case, SRC-ACP can derive the congestion points from the router (see *Deriving Edge Congestion Points* on page 275.)

However, if the access network uses a protocol other than ATM, you must provide the information shown in Table 19 for each subscriber.

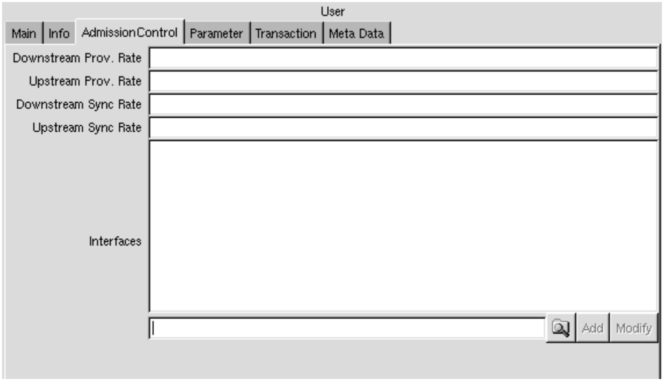
Table 19: SRC-ACP Information for Subscribers

Information	LDAP Attributes
Provisioned downstream bandwidth	downstreamProvisionedRate
Provisioned upstream bandwidth	upstreamProvisionedRate
Actual downstream bandwidth for current subscriber session	downstreamSyncRate
Actual upstream bandwidth for current subscriber session	upstreamSyncRate
List of DNs of interfaces associated with congestion points	networkInterfaceRef

To configure bandwidths for subscribers with SDX Admin:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry for the residential subscriber in *o = users, o = umc*.
3. Click the **Admission Control** tab in the User pane.
4. Enter the values for the fields using the information in *Bandwidth for Subscribers* on page 379.
5. Click **Save** in the User pane.

Figure 46: Admission Control Tab in User Pane



Bandwidth for Subscribers

Use the fields in this section to configure bandwidths for subscribers.

Downstream Prov. Rate

- Provisioned downstream bandwidth.
- Value—Number of bits per second
- Guidelines—Mandatory. This rate is used if the subscriber bandwidth settings are not provided by remote update (through the API for ACP) or by the Downstream Sync Rate value.
- Default—No value

Upstream Prov. Rate

- Provisioned upstream bandwidth.
- Value—Number of bits per second
- Guidelines—Mandatory. This rate is used if the subscriber bandwidth settings are not provided by remote update (through the API for ACP) or by the Upstream Sync Rate value.
- Default—No value

Downstream Sync Rate

- Actual downstream bandwidth for the current subscriber session.
- Value—Number of bits per second
- Guidelines—If you do not set this value and it is not provided by remote update (through the API for ACP), then the Downstream Prov. Rate value is used.
- Default—No value

Upstream Sync Rate

- Actual upstream bandwidth for the current subscriber session.
- Value—Number of bits per second
- Guidelines—If you do not set this value and it is not provided by remote update (through the API for ACP), then the Upstream Prov. Rate value is used.
- Default—No value

Assigning Network Interfaces to Subscribers

You must assign to the subscriber object interfaces (including the router interfaces) for all congestion points between the subscriber and the router. Table 19 on page 378 shows the LDAP attribute for this type of network interface.

NOTE: You must define the interface in the directory before you can assign it to a residential subscriber (see Configuring Network Interfaces in the Directory on page 376).

To assign an interface with SDX Admin:

1. Start at the Admission Control pane for the subscriber (see Figure 46).
2. Click the icon below the Interfaces field.

   The Select Object dialog box appears.
3. Select the network device on which the interface is located.

   You can shift-click or control-click network devices to select multiple options.
4. Click **OK**.

   The network devices appear in the User pane.
5. Click **Add**.

   The network devices appear in the Scopes field of the pane.
6. Highlight a network device.
7. Modify the DN of the network device to include the interface location.
8. Click **Modify**.

Repeat Steps 6 to 8 for each interface associated with a congestion point for this subscriber.

Configuring Bandwidths for Services

Upstream and downstream bandwidths must be specified for services that SRC-ACP manages. You can obtain bandwidths for services in two ways:

- Provide static values through the directory.
- Allow the values to be provided through the SAE core API.

  For example, a business partner may need to specify the required values for a particular piece of content through the SAE core API.

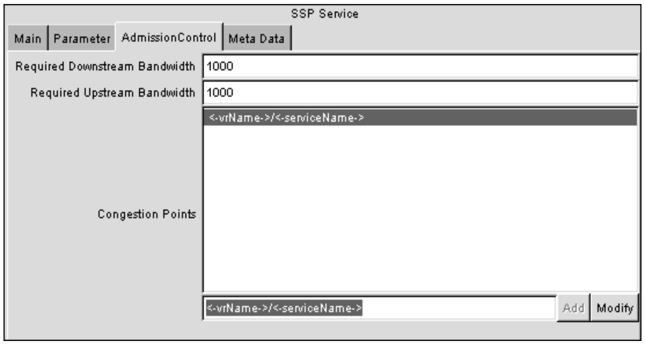
Table 20 shows the LDAP attributes for these services.

Table 20: SRC-ACP Information for Services

Information	LDAP Attributes
Required downstream bandwidth	sspRequiredDownstreamBandwidth
Required upstream bandwidth	sspRequiredUpstreamBandwidth

To configure values for services with SDX Admin:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry for the service in *o = Services, o = umc*.
3. Click the **Admission Control** tab in the SSP Service pane.



Required Downstream Bandwidth: 1000

Required Upstream Bandwidth: 1000

Congestion Points: <-vrName->/<-serviceName->

<-vrName->/<-serviceName->

4. Enter the values for the Required Downstream Bandwidth and the Required Upstream Bandwidth fields.
5. Click **Save** in the SSP Service pane.

Configuring SRC-ACP to Manage the Backbone Network

To configure SRC-ACP to manage the backbone network, you must:

1. Configure network interfaces that represent locations of congestion points in the directory.
2. (Optional) Configure an action congestion point.
3. Configure guaranteed bandwidths for services.
4. Assign network interfaces to services.
5. Create congestion points in the directory.
6. Assign network interfaces to congestion points.

Refer to the following sections for details about these procedures.

NOTE: You can configure objects in the directory by means of an LDAP browser or by means of a network management database. These sections provide information about the LDAP attributes that you must configure and their positions in the LDAP schema, as well as details on how to configure objects with SDX Admin. For detailed information about the LDAP schema, see the documentation in the SRC software distribution in the folder /SDK/doc/ldap.

Configuring Network Interfaces in the Directory

You configure network interfaces in the directory in the same way for edge and backbone congestion points. However, for backbone congestion points, you can add only VRs and their interfaces. For information about this procedure, see *Configuring Network Interfaces in the Directory* on page 376.

Extending SRC-ACP Congestion Points

You can extend SRC-ACP congestion points to initialize and execute applications defined in a backbone congestion point. SRC-ACP provides a service provider interface (SPI) to:

- Create custom congestion point applications that authorize service activation and track service start and stop events.
- Obtain congestion point information from remote update.
- Retrieve congestion point status.
- Track congestion point state.

The SPI for ACP provides a Java interface that a congestion point application implements. For information about the SPI for ACP, see the documentation in the SRC application library distribution in the folder *SDK/doc/acp*.

The implementation of the SPI for ACP can be a customized application that performs certain tasks, such as creating or removing congestion points on the router. SRC-ACP acts as an interface tracking plug-in, and interface tracking events are treated as remote updates for congestion points when they are created, modified, or removed.

SRC-ACP supports applications written in Java or Jython. For scripts written in Java, you must compile and package the implemented SPI for ACP to make it available for use by SRC-ACP. A Java implementation can include more than one Java archive (JAR) file.

To use congestion point applications with SRC-ACP, configure an action congestion point that references the script (see *Configuring Action Congestion Points* on page 383).

Configuring Action Congestion Points

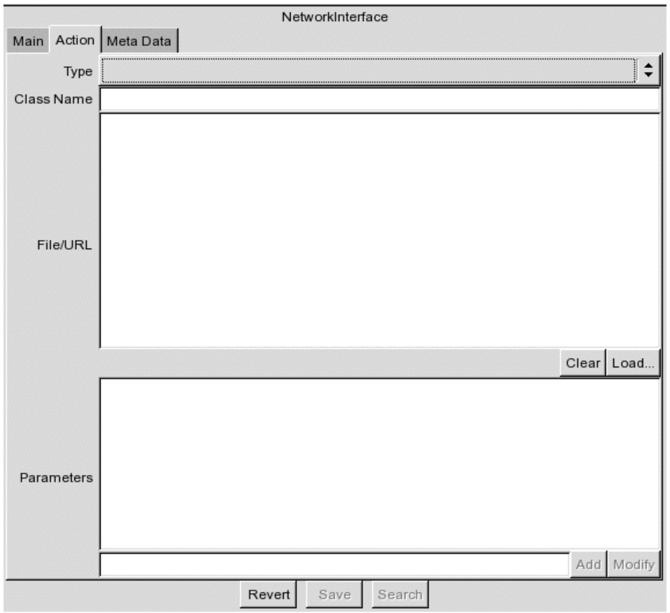
You can define an application in a backbone congestion point so that SRC-ACP can execute it in a predefined manner. Backbone congestion points that are configured to run an application are called action congestion points. If you want to use an action congestion point to execute an application that requires real-time congestion point status, you must enable SRC-ACP state synchronization with the SAE (see *Configuring State Synchronization* on page 368).

Before you configure an action congestion point, make sure that you know the location of the application file.

To configure an action congestion point with SDX Admin:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry for the network interface in *o = Admission Control, o = umc*.
3. Click the **Action** tab.

   The Action tab appears in the NetworkInterface pane.



4. In the Type field, select one of the following file types that the application uses:
   - URL—URL to identify the location of script file
   - Python—Jython source code
   - Java Class—Compiled Java class file
   - Java Archive—Java archive file (.jar)
5. In the Class Name field, enter the class name of the Java or Python class implementing the SPI.
6. In the File/URL field, enter the URL, or click **Load** to add a file.
   - If the type is URL, enter the URL.
   - If the type is not URL, click Load.

     The Load data dialog box appears. For information about loading scripts, see *Managing Files from the Load Data Dialog Box* on page 385.

   The URL or the content of the script file appears in the File/URL box.

   If you want to remove the URL or file, click **Clear**.
7. In the entry box below the Parameters field, enter the parameter as an attribute = value pair, and click **Add**.

   The entry appears in the Parameters field.

To modify the entry:

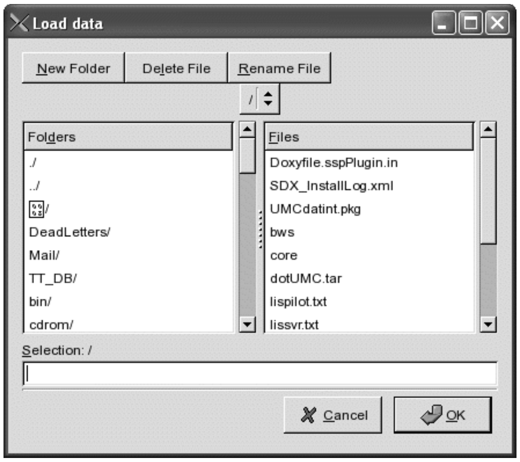
- Highlight the entry so that it appears in the entry box below the Parameters field.
- Make your changes to the entry, and click **Add** to add a new entry, or click **Modify** to change the selected entry.

The modified entry appears in the Parameters field.

8. Click **Save** in the NetworkInterface pane.

Managing Files from the Load Data Dialog Box

If you click **Load** in the Files/URL box, the Load data dialog box appears.



You can manipulate files and folders from the Load data dialog box.

- To load a file:

  1. Select the directory that contains the script that implements the application, and then select the file.

     or

     Type the path to the script file in the Selection box.

     If a JAVA implementation includes more than one JAR file, use commas to separate file URL entries, or enter one URL per line.

  2. Click **OK**.

     The content of the script file appears in the File/URL box.

- To create a new folder, click **New Folder**.
- To remove a file, select a file or enter its path in the Selection box, and click **Delete File**.

- To rename a file:
  1. In the Files list, select a file, and click **Rename File**.

     The Rename File dialog box appears.

  2. Enter the new filename, and click **OK**.

Configuring Bandwidths for Services

You configure bandwidths for services in the same way for edge and backbone congestion points. For information about this procedure, see *Configuring Bandwidths for Services* on page 381.

Configuring Congestion Points for Services

You must assign a congestion point to each service that SRC-ACP manages. Table 21 shows the LDAP attributes for a backbone congestion point.

Table 21: SRC-ACP Information Associated with Backbone Congestion Points

Information	LDAP Attributes
Definition of a backbone congestion point in the format <-vrName->/<-serviceName-> - To allow the software to automatically define the congestion point, use the entry <-vrName->/<-serviceName->. When SRC-ACP starts operating, it will substitute the VR name and the service name from the request for service activation. - To restrict the congestion point to a specific VR or service, enter the actual VR name or service name, as shown in the following examples. - vr1@boston/<-serviceName->—Specifies any service available on VR vr1@boston - <vrName>/news—Specifies the service news on any VR - vr1@boston/news—Specifies the service news available on VR vr1@boston - default@ottawa/news—Specifies the service news available either on the default VR or on a router called ottawa	congestionPoints

To configure values for services with SDX Admin:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry for the service in *o=Services, o=umc*.
3. Click the **Admission Control** tab in the SSP Service pane.
4. In the entry box below the Congestion Points field, enter the name of one congestion point; see Table 21 for information about the format of this entry.
5. Click the **Add** button.

   The entry appears in the Congestion Points field.

Configuring Congestion Points in the Directory

You must add backbone congestion points to the directory in *o = CongestionPoints, o = umc*. Table 21 shows the LDAP attribute for a backbone congestion point.

To configure an individual backbone congestion point in the directory with SDX Admin:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry *o = CongestionPoints, o = umc*, and right-click.
3. Select **New > Congestion Point**.

   The New Congestion Point dialog box appears.
4. Enter the name of the VR that supports the new congestion point, and click **OK**.

   The new object appears in the navigation pane, and basic details for the object appear in the Main tab of the Congestion Point pane.
5. (Optional) Enter a description for the VR in the Description field, and click **Save** in the NetworkDevice pane.
6. In the navigation pane, highlight the VR, and right-click.
7. Select **New > Congestion Point**.

   The New Congestion Point dialog box appears.
8. Enter the name of the service, and click **OK**.

   The new object appears in the navigation pane, and basic details for the object appear in the Main tab of the Congestion Point pane.

To add all backbone congestion points for all VRs in the directory:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry *o = CongestionPoints, o = umc*, and right-click.
3. Select **New > Create CongestionPoints**.

   The VRs and the services they support appear in the folder.

Assigning Interfaces to Congestion Points

You must assign interfaces either to VRs or to individual services under the VRs in *o = CongestionPoints, o = umc*. Services inherit interface assignments from the associated VR unless you assign an interface to the individual service. The LDAP attribute for this network interface is called interfaceRef and it lists the DNs of interfaces associated with backbone congestion points.

To assign interfaces to congestion points with SDX Admin:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry in *o = CongestionPoints, o = umc* to which you want to assign the congestion point.
3. Click the 🔍 icon below the Interfaces field.

   The Select Object dialog box appears.
4. Select the network device on which the interface is located.

   You can shift-click or control-click network devices to select multiple options.
5. Click **OK**.

   The network devices appear in the User pane.
6. Click **Add**.

   The network devices appear in the Scopes field of the pane.

Repeat Steps 3 to 6 for each interface associated with a congestion point for this subscriber.

Defining a Congestion Point Profile

You can create a congestion point profile that automatically performs congestion point classification. This profile supports only access network mode for SRC-ACP.

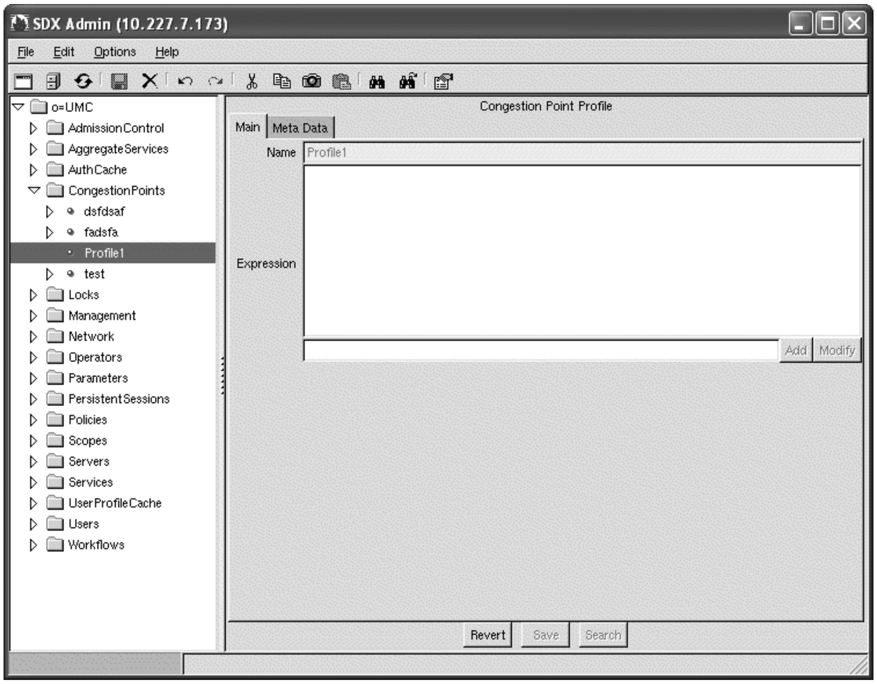
The congestion point profiles are stored in the directory under *o = congestionPoints, o = umc*.

To define a congestion point profile:

1. In SDX Admin under UMC, right-click **CongestionPoints**, select **New**, and then select **Congestion Point Profile**.

   The New Congestion Point Profile dialog box appears.
2. In the New Congestion Point Profile dialog box, enter a name for the profile.

   The Congestion Point Profile pane appears in the content area.



Name Profile1

3. Enter a congestion point expression in the box below the Expression box, and then click **Add**. For information about congestion point expressions, see *Congestion Point Expressions* on page 389.

Congestion Point Expressions

You can enter a congestion point expression by using the syntax listed in this section. You can also embed Python scripting expressions within the congestion point expression.

If you embed Python expressions within a congestion point expression, use the escape sequence <- then -> to enclose the Python expression. See *Methods for Use with Scripting Expressions* on page 390 and *Match Criteria for Congestion Point Classification* on page 391.

The syntax for a congestion point expression is:

<NetworkDevice>/<NetworkInterface>[/<CongestionPoint>]

- <NetworkDevice>—Network device listed in the directory.

  For information about network devices, see *SRC-PE Network Guide, Part 2, Using Juniper Networks Routers in the SRC Network*.

- <NetworkInterface>—Network interface listed in the directory.

  For information about interfaces, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 7, Classifying Interfaces and Subscribers on a Solaris Platform*.

- <CongestionPoint>—(Optional) Name of an instance of congestion point that is automatically created.

If one of the elements with the path contains a slash (/), use a backslash (\) as an escape character for the slash. For example, \/.

Expressions in Templates for Congestion Point Profiles

You can create a congestion point profile to be used as a template for other profiles. Templates simplify management of congestion points. Rather than configuring each congestion point individually, you can create templates to define common parameters for a class of individual congestion points.

For example, in an environment in which VLAN interfaces GigabitEthernet1/0.1 through GigabitEthernet1/0.1000 have the same available bandwidth, you can specify the characteristics of the VLAN interface once and have SRC-ACP create the congestion points based on the template configuration.

When a congestion point expression has the third element, SRC-ACP uses the <NetworkDevice>/<NetworkInterface> part of the expression to load the congestion point from the directory, and uses it as a template to create a congestion point in memory for subscriber. The <CongestionPoint> part of the expression distinguishes each congestion point (available bandwidth) created from this template.

Methods for Use with Scripting Expressions

SRC-ACP provides the following methods to use in scripting expressions:

- slot(nasPortId)—Collects the slot number from the nasPortId or interfaceName

  Example—slot(“atm 4/5:0.32”) == “4”

- port(nasPortId)—Collects the port number from the nasPortId or interfaceName

  Example—port(“atm 4/5:0.32”) == “5”

- l2id(nasPortId)—Collects the layer 2 ID from the nasPortId (VLAN id or ATM vpi.vci)

  Example—l2id(“atm 4/5:0.32”) == “0.32”

- escape(string)—Replaces any slash with the escape sequence \/

  Example—escape("atm 4/5") == "atm 4\/5"

You can extend the scripting library by creating the file *lib/localCPLib.py* in the ACP installation directory, by default */opt/UMC/acp/lib/localCPLib.py*. SRC-ACP reads the definitions in this file at startup, after which they are available for processing.

Match Criteria for Congestion Point Classification

You can use the following match criteria in Python scripting expressions for a congestion point expression:

- ifSessionId—Identifier for RADIUS accounting on the JUNOSe interface.
- authUserId—Identifier that a subscriber uses for authentication.
- domain—Name of the domain used for secondary authentication.
- radiusClass—RADIUS class attribute of the service definition.
- routerName—Name of virtual router in the format <virtualRouter>@<router>.
- interfaceName—Name of the interface, such as fastEthernet3/1.
- interfaceAlias—Alias of the interface; that is, the IP description in the interface configuration.
- interfaceDescr—SNMP description of the interface, such as IP3/1.
- portId— Identifier of VLAN or virtual circuit. For a virtual circuit, use the format <VPI>/<VCI>.
- nasPort—Network access server (NAS) port.
- sspHost—Name of host on which SAE is installed.
- IfRadiusClass—RADIUS class attribute on the JUNOSe interface.
- ServiceBundle—Service bundle vendor-specific attribute for RADIUS. A user authorization plug-in returns this attribute to the SAE.
- loginName—Subscriber's login name. The format of the login name varies.
- primaryUserName—PPP login name or the public DHCP username.
- accountingId—Value of directory attribute accountingUserId.
- userDn—Distinguished name of a subscriber in the directory.
- userMacAddress—Media access controller (MAC) address of a DHCP subscriber.
- dhcpPacket—Content of the DHCP discover request in the format:
  - First 4 octets—Gateway IP address (giaddr field)
  - Remaining octets—DHCP options

  For more information, see RFC 2131—Dynamic Host Configuration Protocol (March 1997) and RFC 2132—DHCP Options and BOOTP Vendor Extensions (March 1997).
- userType—Type of subscriber.

- sessionId—Identifier of RADIUS session for the subscriber session.
- userIp—IP address of the subscriber.
- nasIp—IP address used to communicate with the server.

Examples of Congestion Profiles

This section provides examples of expressions used in congestion point profiles.

Profile for Gigabit Ethernet Interface

In the following example, a subscriber who is connecting to router test@erx through interface GigabitEthernet1/0.1, has the congestion point:

"subInterface=1,interfaceName=GigabitEthernet1/0,orderedCimKeys=test@erx,o=AdmissionControl,o=UMC"

This congestion point is created from a congestion point template:

"interfaceName=GigabitEthernet1/0,orderedCimKeys=test@erx,o=AdmissionControl,o=UMC"

The following congestion point expression is configured for the congestion point profile *cn = vlan, o = CongestionPoints, o = umc*:

NOTE: The following example is a single expression that should be entered on a single line.

```
<-routerName->/<-interfaceName[:interfaceName.find('.')]->/
<-interfaceName[interfaceName.find('.')+1:]->
```

Profile That Contains Three Congestion Points

In the following example, a subscriber who is connecting to router test@erx through interface atm 4/5:0.32 will have three congestion points:

- interfaceName=atm4, orderedCimKeys=
  test@erx, o=AdmissionControl, o=UMC
- interfaceName=atm4/5, orderedCimKeys=test@erx, o=AdmissionControl, o=UMC
- interfaceName=atm4/5:0.32, orderedCimKeys=test@erx, o=AdmissionControl, o=UMC

SRC-ACP automatically appends *o = AdmissionControl, o = UMC*.

This profile creates the same congestion points that are created by the ATM autocompletion feature, which is available in this version as well as previous versions of SRC-ACP.

The following congestion point expressions are configured for the congestion point profile *cn = atm, o = CongestionPoints, o = umc*:

```
<-routerName->/<-escape(portId.replace(‘ ‘,”))->
<-routerName->/<-escape(portId[:portId.rindex(‘.’)].replace(‘ ‘,”))->
<-routerName->/<-escape(portId[:portId.rindex(‘:’)].replace(‘ ‘,”))->
```

Profile That Uses Congestion Point Templates

In the following example, the congestion points are dynamically created based on templates for the expressions. When you use a template, you specify parameters for the < NetworkDevice > / < NetworkInterface > part of the expression. This part refers to a network interface object in the directory that defines the parameters of the congestion point; that is, the available bandwidth.

The following congestion point expressions are configured for the congestion point profile *cn = atm, o = CongestionPoints, o = umc*:

```
<-routerName->/VCI/<-portId->
<-routerName->/VPI/<-portId[:portId.rindex(‘.’)] ->
<-routerName->/PHY/<-portId[:portId.rindex(‘:’)] ->
```

where:

- VCI—Provides parameters for the virtual channel
- VPI—Provides parameters for the virtual path
- PHY—Provides parameters for the physical interface terminating on a JUNOSe router

Changing and Removing a Congestion Point Profile

To change configuration of a congestion point profile:

1. In SDX Admin, select the congestion point profile.
2. In the Congestion Point Profile pane, select the expression to change in the Expression box, and click **Modify**.
3. Make changes to the expression, and click **Add**.

   The updated expression appears in the Expression box.

To remove a congestion point profile:

1. In SDX Admin, right-click the congestion point profile.
2. Select **Delete** to remove the profile.

Starting SRC-ACP

To start SRC-ACP:

1. On the SRC-ACP host, log in as `root` or as an authorized nonroot admin user.
2. Start SRC-ACP from its installation directory.

```
cd /opt/UMC/acp/etc
./acp start
```

Stopping SRC-ACP

To stop SRC-ACP:

1. On the SRC-ACP host, log in as `root` or as an authorized nonroot admin user.
2. Stop SRC-ACP from its installation directory.

```
cd /opt/UMC/acp/etc
./acp stop
```

Monitoring and Managing SRC-ACP

To monitor admission control, you can use the SRC CLI. See *Chapter 24, Monitoring Admission Control with the SRC CLI*.

You can also use the C-Web interface to monitor admission control. See *Chapter 25, Monitoring Admission Control with the C-Web Interface*.

You can use the SRC CLI or the C-Web interface to:

- Display information about the edge network.
- Display information about the backbone network.
- Display information about subscribers and congestions points obtained through an external application.

To manage admission control, you can use the SRC CLI. See *Chapter 23, Managing SRC-ACP with the SRC CLI*.

You can use the SRC CLI to:

- Reorganize the files that contain ACP data.
- Modify congestion points.

Index