



Service Management on SRC-Managed Routers



Modified: 2019-06-21

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Juniper Networks, the Juniper Networks logo, Juniper, and Junos are registered trademarks of Juniper Networks, Inc. in the United States and other countries. All other trademarks, service marks, registered marks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

Service Management on SRC-Managed Routers
Copyright © 2019 Juniper Networks, Inc. All rights reserved.

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at <https://support.juniper.net/support/eula/>. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

	About the Documentation	vii
	Documentation and Release Notes	vii
	Documentation Conventions	vii
	Documentation Conventions	viii
	Documentation Feedback	x
	Requesting Technical Support	xi
	Self-Help Online Tools and Resources	xi
	Creating a Service Request with JTAC	xi
Part 1	Overview	
Chapter 1	Software Features Overview	3
	SRC Component Overview	3
Chapter 2	DPI Script Service	7
	DPI Script Service Overview	7
Part 2	Configuration	
Chapter 3	Configuration Tasks for DPI Script Services	11
	Creating the DPI Script Service (SRC CLI)	11
	Configuring Subscriptions to the DPI Script Service	12
	Parameters for DPI Script Service	15
	Creating a Configuration File	16
	Configuring Batch Parameters	17
	Substituting Parameters in Policy Templates	18
	Configuring Policy Templates	19
	Substituting Parameters in Policy Templates	23
	Configuring Policy Templates	24
Chapter 4	Example	31
	Example: Using the DPI Script Service	31

List of Tables

	About the Documentation	vii
	Table 1: Notice Icons	viii
	Table 2: Notice Icons	ix
	Table 3: Text Conventions	ix
Part 1	Overview	
Chapter 1	Software Features Overview	3
	Table 4: Descriptions of SRC Components	3
Part 2	Configuration	
Chapter 3	Configuration Tasks for DPI Script Services	11
	Table 5: Parameter Substitutions for DPI Services	15
	Table 6: Policy Template Elements for Configuration File	19
	Table 7: Policy Template Elements for Configuration File	25

About the Documentation

- Documentation and Release Notes on page vii
- Documentation Conventions on page vii
- Documentation Feedback on page x
- Requesting Technical Support on page xi

Documentation and Release Notes

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at <https://www.juniper.net/documentation/>.







If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at <https://www.juniper.net/books>.

Documentation Conventions

Table 1 on page viii defines notice icons used in this guide.

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 or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.
	Tip	Indicates helpful information.
	Best practice	Alerts you to a recommended use or implementation.

Documentation Conventions

[Table 1 on page viii](#) defines the notice icons used in this guide. [Table 3 on page ix](#) defines text conventions used throughout this documentation.

Table 2: 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 or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.
	Tip	Indicates helpful information.
	Best practice	Alerts you to a recommended use or implementation.

Table 3: Text Conventions

Convention	Description	Examples
Bold text like this	<ul style="list-style-type: none"> Represents keywords, scripts, and tools in text. Represents a GUI element that the user selects, clicks, checks, or clears. 	<ul style="list-style-type: none"> Specify the keyword exp-msg. Run the install.sh script. Use the pkgadd tool. To cancel the configuration, click Cancel.
Bold text like this	Represents text that the user must type.	user@host# set cache-entry-age <i>cache-entry-age</i>
Fixed-width text like this	Represents information as displayed on your terminal's screen, such as CLI commands in output displays.	<pre> nic-locators { login { resolution { resolver-name /realms/ login/A1; key-type LoginName; value-type SaeId; } } } </pre>
Regular sans serif typeface	<ul style="list-style-type: none"> Represents configuration statements. Indicates SRC CLI commands and options in text. Represents examples in procedures. Represents URLs. 	<ul style="list-style-type: none"> system ldap server{ stand-alone; Use the request sae modify device failover command with the force option user@host# ... https://www.juniper.net/techpubs/software/management/sdx/api-index.html

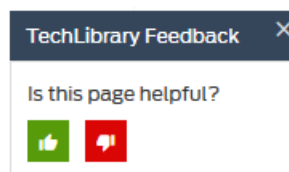
Table 3: Text Conventions (continued)

<i>Italic sans serif typeface</i>	Represents variables in SRC CLI commands.	<code>user@host# set local-address local-address</code>
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.
<i>Italic typeface</i>	<ul style="list-style-type: none"> Emphasizes words. Identifies book names. Identifies distinguished names. Identifies files, directories, and paths in text but not in command examples. 	<ul style="list-style-type: none"> There are two levels of access: <i>user</i> and <i>privileged</i>. <i>SRC-PE Getting Started Guide</i>. <i>o=Users, o=UMC</i> The <i>/etc/default.properties</i> file.
Backslash	At the end of a line, indicates that the text wraps to the next line.	<code>Plugin.radiusAcct-1.class=\ net.juniper.smgmt.sae.plugin\ RadiusTrackingPluginEvent</code>
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.)	<code>diagnostic line</code>

Documentation Feedback

We encourage you to provide feedback so that we can improve our documentation. You can use either of the following methods:

- Online feedback system—Click TechLibrary Feedback, on the lower right of any page on the [Juniper Networks TechLibrary](#) site, and do one of the following:



- Click the thumbs-up icon if the information on the page was helpful to you.
- Click the thumbs-down icon if the information on the page was not helpful to you or if you have suggestions for improvement, and use the pop-up form to provide feedback.
- E-mail—Send your comments to techpubs-comments@juniper.net. Include the document or topic name, URL or page number, and software version (if applicable).

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active Juniper Care or Partner Support Services support contract, or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at <https://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>.
- Product warranties—For product warranty information, visit <https://www.juniper.net/support/warranty/>.
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: <https://www.juniper.net/customers/support/>
- Search for known bugs: <https://prsearch.juniper.net/>
- Find product documentation: <https://www.juniper.net/documentation/>
- Find solutions and answer questions using our Knowledge Base: <https://kb.juniper.net/>
- Download the latest versions of software and review release notes: <https://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications: <https://kb.juniper.net/InfoCenter/>
- Join and participate in the Juniper Networks Community Forum: <https://www.juniper.net/company/communities/>
- Create a service request online: <https://myjuniper.juniper.net>

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: <https://entitlementsearch.juniper.net/entitlementsearch/>

Creating a Service Request with JTAC

You can create a service request with JTAC on the Web or by telephone.

- Visit <https://myjuniper.juniper.net>.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see <https://support.juniper.net/support/requesting-support/>.

PART 1

Overview

- [Software Features Overview on page 3](#)
- [DPI Script Service on page 7](#)

CHAPTER 1

Software Features Overview

- [SRC Component Overview on page 3](#)

SRC Component Overview

The SRC software is a dynamic system. It contains many components that you use to build a subscriber management environment. You can use these tools to customize and extend the SRC software for your use and to integrate the SRC software with other systems. The SRC software also provides the operating system and management tools for C Series Controllers.

[Table 4 on page 3](#) gives a brief description of the components that make up the SRC software.

Table 4: Descriptions of SRC Components

Component	Description
Server Components	
Service activation engine (SAE)	<ul style="list-style-type: none">• Authorizes, activates, and deactivates subscriber and service sessions by interacting with systems such as Juniper Networks routers, cable modem termination system (CMTS) devices, RADIUS servers, and directories.• Collects accounting information about subscribers and services from routers, and stores the information in RADIUS accounting servers, flat files, and other accounting databases.• Provides plug-ins and application programming interfaces (APIs) for starting and stopping subscriber and service sessions and for integrating with systems that authorize subscriber actions and track resource usage.
Subscriber Information Collector (SIC)	The SIC listens for RADIUS accounting events from IP edge devices (accounting clients) and forwards them to a remote AAA server, allowing the SRC software to gain increased subscriber awareness. Additionally, the SIC can optionally edit accounting events before routing them.
Network information collector (NIC)	Collects information about the state of the network and can provide a mapping from a given type of network data to another type of network data.
Redirect Server	Redirects HTTP requests received from IP Filter to a captive portal page.

Table 4: Descriptions of SRC Components (continued)

Component	Description
3GPP Gateway	The SRC Third-Generation Partnership Project (3GPP) gateway is a Diameter-based component in the SRC software, which provides integration with 3GPP Policy and Charging Control environments, to provide fixed-mobile convergence (FMC). The SRC 3GPP gateway provides Gx-based integration with the Policy and Charging Rules Function (PCRF). The SRC 3GPP gateway uses the northbound Gx interface to mediate between the PCRF and Juniper Networks routers like the E Series Broadband Services routers and MX Series routers. The northbound Gx interface on the SRC 3GPP gateway communicates with the PCRF using the Diameter protocol.
3GPP Gy	The SRC 3GPP Gy is a Diameter-based component in the SRC software, which provides Gy-based integration with the Online Charging System (OCS), to provide FMC. The SRC 3GPP Gy uses the northbound Gy interface to handle charging-related information between the OCS and Juniper Networks routers like the E Series Broadband Services routers and MX Series routers. The northbound Gy interface communicates with the OCS using the Diameter protocol.
Web Application Service	The SRC software includes a Web application server that hosts the Web Services Gateway and the Volume Tracking Application (SRC VTA). In production environments, this application server is designed to host only these applications. However, you can load your own applications into this server for testing or demonstration purposes.
Web Services Gateway	Allows a gateway client—an application that is not part of the SRC network—to interact with SRC components through a Simple Object Access Protocol (SOAP) interface. The Web Services Gateway provides the Dynamic Service Activator which allows a gateway client to dynamically activate and deactivate SRC services for subscribers and to run scripts that manage the SAE.
Monitor Components Connectivity (MCC)	Monitors the connectivity state between SAEs in a community and between SAE and RADIUS server periodically and collects diagnostic information about the connectivity state of components, such as connection error, connection timeout, and socket read/write timeout.
Repository	
Directory	The SRC software includes the Juniper Networks database, which is a built-in Lightweight Directory Access Protocol (LDAP) directory for storing all SRC data including services, policies, and small subscriber databases. For large subscriber databases, you must supply your own directory.
SRC Configuration and Management Tools	
SRC command line interface (CLI)	Provides a way to configure the SRC software on a C Series Controller from a Junos OS–like CLI. The SRC CLI includes the policies, services, and subscribers CLI, which has separate access privileges.
C-Web interface	Provides a way to configure, monitor, and manage the SRC software on a C Series Controller through a Web browser. The C-Web interface includes a policies, services, and subscribers component, which has separate access privileges.
Simple Network Management Protocol (SNMP) agent	Monitors system performance and availability. It runs on all the SRC hosts and makes management information available through SNMP tables and sends notifications by means of SNMP traps.

Table 4: Descriptions of SRC Components (continued)

Component	Description
Service Management Applications (Run on external system)	
IMS Services Gateway	Integrates into an IP multimedia system (IMS) environment. The SRC software provides a Diameter protocol-based interface that allows the SRC software to integrate with services found on the application layer of IMS.
SRC Programming Interfaces	
NETCONF API	Allows you to configure or request information from the NETCONF server on a C Series Controller that runs the SRC software. Applications developed with the NETCONF API run on a system other than a C Series Controller.
CORBA plug-in service provider interface (SPI)	Tracks sessions and enables linking the rest of the service provider's operations support system (OSS) with the SRC software so that the OSS can be notified of events in the life cycle of SAE sessions. Hosted plug-ins only.
CORBA remote API	Provides remote access to the SAE core API. Applications that use these extensions to the SRC software run on a system other than a C Series Controller.
NIC access API	Performs NIC resolutions. Applications that use these extensions to the SRC software run on a system other than a C Series Controller.
SAE core API	Controls the behavior of the SRC software. Applications that use these extensions to the SRC software run on a system other than a C Series Controller.
Script services	Provides an interface to call scripts that supply custom services such as provisioning policies on a number of systems across a network.
VTA API	The Volume Tracking Application (VTA) API is a Simple Object Access Protocol (SOAP) interface that allows developers to create gateway clients and that administrators use to manage VTA subscribers and sessions. The SRC Web Services Gateway allows a gateway client—an application that is not part of the SRC network—to interact with SRC components, such as the VTA, through a SOAP interface.
Authorization and Accounting Applications	
AAA RADIUS servers	Authenticates subscribers and authorizes their access to the requested system or service. Accepts accounting data—time active and volume of data sent—about subscriber and service sessions. RADIUS servers run on a system other than a C Series Controller.
SRC Admission Control Plug-In (SRC ACP)	Authorizes and tracks subscribers' use of network resources associated with services that the SRC application manages.
Flat file accounting	Stores tracking data to accounting flat files that can be made available to external systems that send the data to a rating and billing system.

Table 4: Descriptions of SRC Components (continued)

Component	Description
Volume Tracking Application	<p>The SRC Volume Tracking Application (SRC VTA) is an SRC component that allows service providers to track and control the network usage of subscribers and services. You can control volume and time usage on a per-subscriber or per-service basis. This level of control means that service providers can offer tiered services that use volume as a metric, while also controlling abusive subscribers and applications.</p> <p>When a subscriber or service exceeds bandwidth limits (or quotas), the SRC VTA can take actions including imposing rate limits on traffic, sending an e-mail notification, or charging extra for additional bandwidth consumed.</p>
Demonstration Applications (available on the Juniper Networks Website)	
Enterprise Audit Plug-In	Defines a callback interface, which receives events when IT managers complete specified operations.
Enterprise Manager Portal	<p>Allows service providers to provision services for enterprise subscribers on routers running JunosE or Junos OS and allows IT managers to manage services.</p> <p>Enterprise Manager Portal can be used with NAT Address Management Portal to allow service providers to manage public IP addresses for use with NAT services on routers running Junos OS and to all IT managers to make requests about public IP addresses through the Enterprise Manager Portal.</p>
Monitoring Agent application	Integrates IP address managers, such as a DHCP server or a RADIUS server, into an SRC-managed network so that the SAE is notified about subscriber events. The Monitoring Agent application runs on a Solaris platform.
Residential service selection portals	Provides a framework for building Web applications that allow residential and enterprise subscribers to manage their own network services. It comes with several full-featured sample Web applications that are easy to customize and suitable for deployment. The Residential service selection portals run on a Solaris platform.
Sample enterprise service portal	Lets service providers supply an interface to their business customers for managing and provisioning services.

Related Documentation • *SRC Product Description*

CHAPTER 2

DPI Script Service

- [DPI Script Service Overview on page 7](#)

DPI Script Service Overview

The SRC software has a Deep Packet Inspection (DPI) script service that can be used to modify services on routers managed by the SRC software to provide resource management based on supported applications. The DPI script service allows the SRC software to manage services on Juniper Networks routers running Junos OS, such as the Juniper Networks MX Series Ethernet Services Router. When the service activation engine (SAE) activates the DPI script service, the session uses the command channel to manage policies on MX Series routers. The DPI script service can use the Junos XML management protocol to change the configuration on the MX Series router as well as send commands to third-party devices that support Telnet access.

The DPI script service activates services by applying policy rules to the managed interface on a network device. The DPI script service allows you to modify existing rules by parameter substitutions; that is, you can replace existing rules with new rules or delete existing rules. The DPI script service supports only default SRC subscriptions and default SAE service sessions. The script service supports policy rules to handle Junos XML management protocol requests for MX Series routers and Telnet commands for third-party routers.

Related Documentation

- For information about the Junos XML management protocol, see the *Junos XML Management Protocol Developer Guide*
- For information about service set configuration on routers running Junos OS, see the *Junos OS Services Interfaces Library for Routing Devices*
- For information about accessing and configuring the third-party device using Telnet, see the device's software documentation

PART 2

Configuration

- [Configuration Tasks for DPI Script Services on page 11](#)
- [Example on page 31](#)

CHAPTER 3

Configuration Tasks for DPI Script Services

- [Creating the DPI Script Service \(SRC CLI\) on page 11](#)
- [Configuring Subscriptions to the DPI Script Service on page 12](#)
- [Parameters for DPI Script Service on page 15](#)
- [Creating a Configuration File on page 16](#)
- [Substituting Parameters in Policy Templates on page 23](#)
- [Configuring Policy Templates on page 24](#)

Creating the DPI Script Service (SRC CLI)

To create the script service:

1. Create a script service in the [**edit services scope *name* service *name***] hierarchy. In this sample procedure, the service is configured in the DPI service scope, and DPI is the name of the service.

```
user@host# edit services scope DPI service DPI
```

2. Set the type of service to script.

```
[edit services scope DPI service DPI]  
user@host# set type script
```

3. (Optional) Configure other properties as needed for your service.
4. Configure the script properties.

```
[edit services scope DPI service DPI]  
user@host# edit script
```

5. Configure *net.juniper.smgtpiscriptservice.DpiService* as the name of the class that implements the script service.

```
[edit services scope DPI service DPI script]  
user@host# set class-name net.juniper.smgtpiscriptservice.DpiScriptService
```

6. Configure Java archive file as the type of script that the script service uses.

```
[edit services scope DPI service DPI script]
user@host# set script-type java-archive
```

7. Specify the filename of the script service implementation so that its contents will be loaded into the **file** option. Copy the *dpiss.jar* file to the C Series Controller before you specify the filename. In this sample procedure, the *dpiss.jar* file was copied from a location that is accessible by a URL (such as an FTP or HTTP server) to the */tmp* directory.

```
[edit services scope DPI service DPI script]
user@host# run file copy URL /tmp/dpiss.jar
user@host# set filename /tmp/dpiss.jar
```

8. (Optional) From configuration mode, enter the service parameter configuration and configure values for parameters.

```
user@host# edit services scope DPI service DPI parameter
```

```
[edit services scope DPI service DPI parameter]
user@host# set substitution [substitution...]
```

For example, to specify the configuration file that is used or to specify whether the configuration file is reloaded for each script service activation:

```
user@host# set substitution dpiConfig=\"resource/dpiConf.xml\"
user@host# set substitution dpiConfigDebug=\"off\"
```

9. (Optional) Verify your configuration.

```
[edit services scope DPI service DPI]
user@host# show
```

Related Documentation

- [SRC Script Services Overview](#)

Configuring Subscriptions to the DPI Script Service

You configure subscriptions to the DPI script service by adding subscribers. You can set up the subscription to activate immediately on login.

To add access subscribers:

1. From configuration mode, enter the **[edit subscribers retailer]** hierarchy. In this sample procedure, the retailer called DPI is configured in the DPI service scope.

```
user@host# edit subscribers retailer DPI
```

2. Specify the domain name associated with the retailer.

```
[edit subscribers retailer DPI]  
user@host# set domain-name [domain-name...]
```

3. (Optional) Assign service scopes for the retailer.

```
[edit subscribers retailer DPI]  
user@host# set scope [scope...]
```

4. Add a subscriber folder for the retailer. In this sample procedure, local is the name of the subscriber folder.

```
[edit subscribers retailer DPI]  
user@host# edit subscriber-folder local
```

5. Add an enterprise subscriber. In this sample procedure, ENT is the name of the enterprise subscriber.

```
[edit subscribers retailer DPI subscriber-folder local]  
user@host# edit enterprise ENT
```

6. (Optional) Configure values for parameters.

```
[edit subscribers retailer DPI subscriber-folder local enterprise ENT]  
user@host# set substitution [substitution...]
```

For example, to specify the dpiRules or dpiAdminState parameter:

```
user@host# set substitution 'dpiRules=[{app="rtsp", action="accept",  
fcl="expedited-forwarding"}, {app="bittorrent", action="discard"}]'  
user@host# set substitution dpiAdminState=\"enabled\"
```

7. Configure accesses for the enterprise subscriber.

```
[edit subscribers retailer DPI subscriber-folder local enterprise ENT]
user@host# edit access name
```

8. Specify the interface name associated with the access using the CLI syntax of the device.

```
[edit subscribers retailer DPI subscriber-folder local enterprise ENT access name]
user@host# set interface-name interface-name
```

9. (Optional) Configure actual values for parameters.

```
[edit subscribers retailer DPI subscriber-folder local enterprise ENT access name]
user@host# set substitution [substitution...]
```

For example, to specify the interface class:

```
user@host# set substitution 'dpiInterfaceClasses=["MXEnterprise"]'
```

10. From configuration mode, enter the subscription configuration. In this sample procedure, the service called DPI is configured for the enterprise.

```
user@host# edit subscribers retailer DPI subscriber-folder local enterprise ENT
subscription DPI
```

11. Specify that the service is activated on login.

```
[edit subscribers retailer DPI subscriber-folder local enterprise ENT subscription DPI]
user@host# set activation automatically-on-login
```

Related Documentation

- *Adding Subscribers (SRC CLI)*
- *Adding Retailers (SRC CLI)*
- *Adding Subscriber Folders (SRC CLI)*
- *Adding Enterprises (SRC CLI)*
- *Configuring Accesses (SRC CLI)*
- *Configuring Subscriptions (SRC CLI)*
- *Configuring Subscribers and Subscriptions Overview*

Parameters for DPI Script Service

Table 5 on page 15 lists the parameters specified by the DPI script service, which is implemented by the `/SDK/scriptServices/dpiScriptService/lib/dpiss.jar` file found in the `SDK+AppSupport+Demos+Samples.tar.gz` file. The value assigned to the parameter must be enclosed by quotation marks. For example, `dpiConfig="resource/dpiConf.xml"` specifies the pathname for the configuration file.

Table 5: Parameter Substitutions for DPI Services

Parameter Name	Description
dpiConfig	<p>Configuration file in the format of a URL or pathname. The script service handles each format as follows:</p> <ul style="list-style-type: none"> URL—String that starts with http:. The script service uses the value to download the configuration file. pathname—String that does not start with http:. The script service uses the value as a path to a resource in the <code>.jar</code> file that contains the script service. <p>We recommend using a pathname in a production environment.</p> <p>If you do not supply a value, the default value is <code>"resource/dpiConf.xml"</code>.</p>
dpiConfigDebug	<p>Reloads the configuration file specified by dpiConfig for each script service activation. Specify "on" to perform the reload. The default value is "off", where the script service uses the configuration file that is accessed for the first service activation until the SAE is restarted.</p>
dpiAdminState	<p>Used by the application to activate the service. By default, the value is set to "enabled". Set the value to "disabled" to deactivate the service.</p> <p>NOTE: If you set the value to disabled, the service session continues to exist because the service is not deactivated. This behavior allows the application to monitor the operational state of the service so that the application knows when a configuration change has committed.</p>
dpiOprState	<p>Used by the application to determine the operational state of the service.</p> <p>When you activate, deactivate, or change the parameters of the DPI script service, the operational state is set to commit pending. Once the commit succeeds, the operational state changes to committed.</p> <p>NOTE: The <code>dpiOprState</code> parameter is reserved for the use of the DPI script service. Another application, such as an enterprise Web application, cannot use this parameter.</p>

Table 5: Parameter Substitutions for DPI Services (continued)

Parameter Name	Description
dpiMaintMode	<p>Prevents the script service from making configuration changes. You can set this parameter in a service scope attached to a specific router so that it affects only that router. If you set this parameter in the service, the whole network is put in maintenance mode.</p> <p>Specify "off" (the default value) to turn off maintenance mode.</p> <p>You can specify "on" to allow the script service to continue recording configuration changes and to maintain the corresponding service sessions in the commit pending state until maintenance mode is set to "off".</p>
dpiRules	<p>Defines application rules as a list of map expressions that the policy template can use to bind variables in the policy template to values in this list. See the <for-each-rule> element in the configuration file.</p> <p>The keys in the map expression must be valid identifiers for substitutions. For example, four or fewer hexadecimal digits are interpreted as parts of IPv6 addresses and cannot be used as identifiers.</p> <p>If you do not supply a value, the default value is an empty list.</p>
dpiInterfaceClasses	<p>Specifies the policy templates that should be applied to the interface. See the <target> element in the configuration file.</p> <p>This parameter lets you group your interfaces into classes so that you can specify for each interface which targets should be applied when the DPI script service is triggered. For example, you might have some interfaces with services provided entirely by an MX Series router and some interfaces that have a customer premises router. In your DPI script service configuration, you can have one target to configure the MX Series router with Junos XML management protocol commands and another target to configure the CPE router with Telnet commands.</p> <p>The interface is normally specified as a list of strings in the subscriber hierarchy to define the interfaces affected by the service activation. If you do not supply a value, the default value is an empty list.</p>

Related Documentation

- [Creating the DPI Script Service \(SRC CLI\) on page 11](#)
- [Configuring Subscriptions to the DPI Script Service on page 12](#)
- [Substituting Parameters in Policy Templates on page 18](#)
- [Configuring Policy Templates on page 19](#)

Creating a Configuration File

The `/SDK/scriptServices/dpiScriptService/resource/dpiConf.xml` file found in the `SDK+AppSupport+Demos+Samples.tar.gz` file contains a sample configuration file for

the script service that demonstrates service activation and deactivation using Junos XML management protocol and Telnet commands.

The configuration file is in the form of an XML document with these sections:

- Optimization parameters—Optional section to optimize batch parameters for committing configuration changes
- Policy templates—Mandatory section for specifying the policy rules that will be added to or removed from network devices during service activation or deactivation

The configuration file has this basic structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE dpi-configuration SYSTEM "dpi-configuration.dtd" >
<configuration>
  <batch <!-- batch parameters specified as attributes --> />
  <policy-templates>
    <!-- policy template specification -->
  </policy-templates>
</configuration>
```

Tasks to create a configuration file are:

- [Configuring Batch Parameters on page 17](#)
- [Substituting Parameters in Policy Templates on page 18](#)
- [Configuring Policy Templates on page 19](#)

Configuring Batch Parameters

To avoid the overhead of individual commits for policy changes that occur close together, the script service groups the policy changes for a network device into a batch so that the commits can happen at the same time.

To optimize the batch parameters, you can specify these timing attributes for the **<batch/>** element:

- **wait-time**—Time to wait for the next command for each device before committing the configuration. The default value is 60 seconds.
- **max-commit-delay**—Maximum time to wait before committing the configuration. The default value is 120 seconds.

For example, to specify a **wait-time** of 15 seconds and **max-commit-delay** of 30 seconds:

```
<batch wait-time="15" max-commit-delay="30" />
```

- See Also**
- [Substituting Parameters in Policy Templates on page 18](#)
 - [Configuring Policy Templates on page 19](#)

Substituting Parameters in Policy Templates

The SRC software can substitute values for variables in the policy templates. The following variables are supported in the policy templates:

- Attributes of the ServiceSessionInfo object in the SAE script service API
- Variables from parameter acquisition
- Variables defined in the dpiRules parameter that are acquired in the context of a **<for-each-rule>** element



NOTE: If the same variable is defined in both the ServiceSessionInfo interface and parameter acquisition, the value in the ServiceSessionInfo interface is used. However, the value defined in the dpiRules parameter override the other values.

For information about the ServiceSessionInfo interface, see the script service documentation in the SAE core API documentation on the Juniper Networks website at <https://www.juniper.net/documentation/software/management/src/api-index.html>.

The value of the variable can be used in the policy templates as defined. You can also specify how to use the value of a variable by extracting part of the value from the variable or replacing nonalphanumeric characters in the value with underscores.

- To extract part of the value from the variable, follow the variable with a tilde (~) and a Java regular expression pattern. The regular expression is matched against the value of the variable, and the value of the last capture group is the result of instantiating the variable expression. For more information about using regular expressions, see <http://java.sun.com/javase/6/docs/api/java/util/regex/Pattern.html>.

For example: `[[[variable~[^\.]+\.(\\d+)]]]`

If we replace *variable* with *interfaceName* and the value of *interfaceName* is *ge-1/2/3.4*, then this expression would evaluate to *4*.

- To replace all nonalphanumeric characters in the value with underscores, follow the variable with an underscore (_).

For example: `[[[variable_]]]`

If we replace *variable* with *interfaceName* and the value of *interfaceName* is *ge-1/2/3.4*, then this expression would evaluate to *ge_1_2_3_4*.



NOTE: You can use the underscore and the tilde expressions together, but the underscore must precede the tilde in the expression.

See Also • [Configuring Policy Templates on page 19](#)

Configuring Policy Templates

The policy templates are used to define the policy rules that are inserted or removed from network devices. Templates are combined with parameters from the service activation context to generate Junos XML management protocol and Telnet commands that add and remove service policies.

The policy templates section has this basic structure:

```
<policy-templates>
  <target interface-class="<!-- interface class name -->">
    <activation>
      <junoscript>
        <!-- JUNOScript API statements -->
        <for-each-rule>
          <!-- Can have multiple for-each-rule -->
          <if test="expression">
            <!-- Can have conditional expressions -->
          </if>
        </for-each-rule>
        <for-each-rule test="expression">
          <!-- Can have multiple for-each-rule -->
          <!-- For each single rule, can include test conditions -->
        </for-each-rule>
      </junoscript>
      <telnet host="<!-- hostname -->">
        <prompt>login:</prompt>
        <command>joe</command>
        <prompt>password:</prompt>
        <command>abc123</command>
        <!-- Can have many prompt/command pairs -->
        <for-each-rule test="expression">
          <!-- For each single rule, can include conditions
            and have prompt/command pairs -->
        </for-each-rule>
      </telnet>
    </activation>
    <deactivation>
      <!-- Structure same as for activation -->
    </deactivation>
  </target>
</policy-templates>
```

Table 6 on page 19 describes the policy template elements in the configuration file.

Table 6: Policy Template Elements for Configuration File

Element	Description
<target interface-class="interface-class-name">	<p>Defines a single policy template, which is selected by matching the interface-class attribute with the value found in the <code>dpiInterfaceClasses</code> parameter. If the interface-class attribute is not provided or its value is "", the target applies to all interfaces.</p> <p>For example: <target interface-class="MXEnterprise"></p>

Table 6: Policy Template Elements for Configuration File (continued)

Element	Description
<activation>	Defines what the script service should do when activating or modifying a session. This element is triggered when the dpiAdminState parameter changes from "disabled" to "enabled".
<deactivation>	Defines what the script service should do when deactivating a session. This element is triggered when the dpiAdminState parameter changes from "enabled" to "disabled".
<junoscript>	<p>Contains a sequence of Junos XML management protocol commands to manage policies on routers running Junos OS.</p> <p>This element can contain <if> and <for-each-rule> elements, delimited variables, literal text, and XML elements, which are not interpreted.</p>
<telnet host="hostname">	<p>Contains a sequence of prompt and command pairs to match on the Telnet device, similar to an expect script. The host attribute is a variable that can include a regular expression to extract a part of the value from the variable. See the <variable> element.</p> <p>For example: <telnet host="deviceIP"></p> <p>This element can contain <if>, <for-each-rule>, <prompt>, and <command> elements. The <prompt> and <command> elements must alternate, and the sequence must start with the <prompt> element. This element can also contain delimited variables and literal text.</p>
<variable-delimiters start="delimiter" end="delimiter">	<p>Specifies the delimiters for variables in the configuration file. The default delimiters enclose the variable within three square brackets ([[[variable]]]).</p> <p>If you want to specify a different delimiter, you must specify the <variable-delimiters> element immediately after the opening tag for the <junoscript> or <telnet> element. The delimiters apply to the contents of the <junoscript> or <telnet> element. Any other occurrences of the <variable-delimiters> element within that element are ignored.</p> <p>For example: <variable-delimiters start="*" end="*"></p>
<if test= "variableName~pattern">	<p>Defines conditional expressions used to generate configuration commands.</p> <p>The test attribute is a variable expression without delimiters. The test is true if the variable has a value and if the optional regular expression matches the variable.</p> <p>For example, the forwarding-class statement would be added to the body only if the map expression contained the fcl key to satisfy the test condition:</p> <pre><if test="fcl"> forwarding-class [[[fcl]]]; </if></pre>

Table 6: Policy Template Elements for Configuration File (continued)

Element	Description
<for-each-rule>	<p>Creates the specified body in the policy template for instantiating each map expression found in the <code>dpiRules</code> parameter. For example, if you have two map expressions in the <code>dpiRules</code> parameter, the policy template would generate the body of the <for-each-rule> element once for each map expression.</p> <p>The <for-each-rule> element has a <code>ruleNumber</code> variable to sequentially track the processing of each map expression.</p> <p>You can use the test attribute to provide a condition for the rule; using this attribute would be the same as adding an <if> element.</p>



NOTE: When using special XML characters as part of the policy templates, they must be coded in XML. For example, the left angle bracket (<) must be coded as `<`.

The following example uses some elements to show a policy template that activates application-aware access list (AACL) services and service sets on an MX Series router by loading the configuration in text format using Junos XML management protocol.

```

<policy-templates>
  <target interface-class="MXEnterprise">
    <activation>
      <junoscript>
        <rpc>
          <load-configuration action="replace" format="text">
            <configuration-text>
services {
  aacl {
    rule AACL_{{{ interfaceName_ }}} {
      match-direction input-output;
      <for-each-rule>
        term {{{ ruleNumber }}} {
          from {
            application junos:{{{ app }}};
          }
          then {
            <if test="fcl">
              forwarding-class {{{ fcl }}};
            </if>
            <if test="action~accept">
              count application;
            </if>
            {{{ action }}};
          }
        }
      </for-each-rule>
    }
  }
  service-set SSET_{{{ interfaceName_ }}} {
    aacl-rules AACL_{{{ interfaceName_ }}};
  }
}
            
```

```

        interface-service {
            service-interface ms-1/[[[ interfaceName~[^\.]+\d+(\d+/\d+/\d+.\d+)
]]];
        }
    }
}
interfaces {
    [[[ interfaceName~[^\.]+\d+ ]]] {
        unit [[[ interfaceName~[^\.]+\d+(\d+) ]]] {
            family inet {
                service {
                    input {
                        service-set SSET_[[[ interfaceName_ ]]]
                    }
                    output {
                        service-set SSET_[[[ interfaceName_ ]]]
                    }
                }
            }
        }
    }
}
}
</configuration-text>
</load-configuration>
</rpc>
</junoscript>
</activation>
</target>
</policy-templates>

```

If the example uses the following dpiRules substitution:

```

dpiRules=[{app="rtsp", action="accept", fcl="expedited-forwarding"},
{app="bittorrent", action="discard"}]

```

The two map expressions in the dpiRules parameter might generate the following target configuration (with two terms) from the policy template example:

```

services {
    aacl {
        rule AAACL_xe_8_3_0_1001 {
            match-direction input-output;
            term 1 {
                from {
                    applications junos:rtsp;
                }
                then {
                    forwarding-class expedited-forwarding;
                    count application;
                    accept;
                }
            }
            term 2 {
                from {
                    applications junos:bittorrent;
                }
                then {

```

```

        discard;
    }
}
}
service-set SSET_xe_8_3_0_1001 {
    aacl-rules AACL_{{{ interfaceName_ }}};
    interface-service {
        service-interface ms-1/3/0.1001;
    }
}
}
interfaces {
    xe-8/3/0 {
        unit 1001 {
            family inet {
                service {
                    input {
                        service-set SSET_xe_8_3_0_1001;
                    }
                    output {
                        service-set SSET_xe_8_3_0_1001;
                    }
                }
            }
        }
    }
}
}
}

```

See Also • [Substituting Parameters in Policy Templates on page 18](#)

- For information about the Junos XML management protocol, see *Junos XML Management Protocol Developer Guide*
- For information about service set configuration on routers running Junos OS, see *Junos OS Services Interfaces Library for Routing Devices*
- For information about accessing and configuring the third-party device using Telnet, see the device's software documentation

Related Documentation

- See *Junos XML Management Protocol Developer Guide*
- See *Junos OS Services Interfaces Library for Routing Devices*
- [Creating the DPI Script Service \(SRC CLI\) on page 11](#)

Substituting Parameters in Policy Templates

The SRC software can substitute values for variables in the policy templates. The following variables are supported in the policy templates:

- Attributes of the ServiceSessionInfo object in the SAE script service API

- Variables from parameter acquisition
- Variables defined in the dpiRules parameter that are acquired in the context of a `<for-each-rule>` element



NOTE: If the same variable is defined in both the ServiceSessionInfo interface and parameter acquisition, the value in the ServiceSessionInfo interface is used. However, the value defined in the dpiRules parameter override the other values.

For information about the ServiceSessionInfo interface, see the script service documentation in the SAE core API documentation on the Juniper Networks website at <https://www.juniper.net/documentation/software/management/src/api-index.html>.

The value of the variable can be used in the policy templates as defined. You can also specify how to use the value of a variable by extracting part of the value from the variable or replacing nonalphanumeric characters in the value with underscores.

- To extract part of the value from the variable, follow the variable with a tilde (~) and a Java regular expression pattern. The regular expression is matched against the value of the variable, and the value of the last capture group is the result of instantiating the variable expression. For more information about using regular expressions, see <http://java.sun.com/javase/6/docs/api/java/util/regex/Pattern.html>.

For example: `[[[variable~[^\.]+\.(\\d+)]]]`

If we replace *variable* with *interfaceName* and the value of *interfaceName* is *ge-1/2/3.4*, then this expression would evaluate to *4*.

- To replace all nonalphanumeric characters in the value with underscores, follow the variable with an underscore (_).

For example: `[[[variable_]]]`

If we replace *variable* with *interfaceName* and the value of *interfaceName* is *ge-1/2/3.4*, then this expression would evaluate to *ge_1_2_3_4*.



NOTE: You can use the underscore and the tilde expressions together, but the underscore must precede the tilde in the expression.

Related Documentation

- [Configuring Policy Templates on page 19](#)

Configuring Policy Templates

The policy templates are used to define the policy rules that are inserted or removed from network devices. Templates are combined with parameters from the service activation context to generate Junos XML management protocol and Telnet commands that add and remove service policies.

The policy templates section has this basic structure:

```
<policy-templates>
  <target interface-class="<!-- interface class name -->">
    <activation>
      <junoscript>
        <!-- JUNOScript API statements -->
        <for-each-rule>
          <!-- Can have multiple for-each-rule -->
          <if test="expression">
            <!-- Can have conditional expressions -->
          </if>
        </for-each-rule>
        <for-each-rule test="expression">
          <!-- Can have multiple for-each-rule -->
          <!-- For each single rule, can include test conditions -->
        </for-each-rule>
      </junoscript>
      <telnet host="<!-- hostname -->">
        <prompt>login:</prompt>
        <command>joe</command>
        <prompt>password:</prompt>
        <command>abc123</command>
        <!-- Can have many prompt/command pairs -->
        <for-each-rule test="expression">
          <!-- For each single rule, can include conditions
            and have prompt/command pairs -->
        </for-each-rule>
      </telnet>
    </activation>
    <deactivation>
      <!-- Structure same as for activation -->
    </deactivation>
  </target>
</policy-templates>
```

Table 6 on page 19 describes the policy template elements in the configuration file.

Table 7: Policy Template Elements for Configuration File

Element	Description
<target interface-class="interface-class-name">	<p>Defines a single policy template, which is selected by matching the interface-class attribute with the value found in the <code>dpiInterfaceClasses</code> parameter. If the interface-class attribute is not provided or its value is "", the target applies to all interfaces.</p> <p>For example: <target interface-class="MXEnterprise"></p>
<activation>	<p>Defines what the script service should do when activating or modifying a session. This element is triggered when the <code>dpiAdminState</code> parameter changes from "disabled" to "enabled".</p>
<deactivation>	<p>Defines what the script service should do when deactivating a session. This element is triggered when the <code>dpiAdminState</code> parameter changes from "enabled" to "disabled".</p>

Table 7: Policy Template Elements for Configuration File (continued)

Element	Description
<code><junoscript></code>	<p>Contains a sequence of Junos XML management protocol commands to manage policies on routers running Junos OS.</p> <p>This element can contain <code><if></code> and <code><for-each-rule></code> elements, delimited variables, literal text, and XML elements, which are not interpreted.</p>
<code><telnet host="hostname"></code>	<p>Contains a sequence of prompt and command pairs to match on the Telnet device, similar to an expect script. The <code>host</code> attribute is a variable that can include a regular expression to extract a part of the value from the variable. See the <code><variable></code> element.</p> <p>For example: <code><telnet host="deviceIP"></code></p> <p>This element can contain <code><if></code>, <code><for-each-rule></code>, <code><prompt></code>, and <code><command></code> elements. The <code><prompt></code> and <code><command></code> elements must alternate, and the sequence must start with the <code><prompt></code> element. This element can also contain delimited variables and literal text.</p>
<code><variable-delimiters start="delimiter" end="delimiter"></code>	<p>Specifies the delimiters for variables in the configuration file. The default delimiters enclose the variable within three square brackets (<code>[[[variable]]]</code>).</p> <p>If you want to specify a different delimiter, you must specify the <code><variable-delimiters></code> element immediately after the opening tag for the <code><junoscript></code> or <code><telnet></code> element. The delimiters apply to the contents of the <code><junoscript></code> or <code><telnet></code> element. Any other occurrences of the <code><variable-delimiters></code> element within that element are ignored.</p> <p>For example: <code><variable-delimiters start="(*" end="*)"></code></p>
<code><if test= "variableName~pattern"></code>	<p>Defines conditional expressions used to generate configuration commands.</p> <p>The <code>test</code> attribute is a variable expression without delimiters. The test is true if the variable has a value and if the optional regular expression matches the variable.</p> <p>For example, the forwarding-class statement would be added to the body only if the map expression contained the fcl key to satisfy the test condition:</p> <pre><if test="fcl"> forwarding-class [[[fcl]]]; </if></pre>

Table 7: Policy Template Elements for Configuration File (continued)

Element	Description
<for-each-rule>	<p>Creates the specified body in the policy template for instantiating each map expression found in the <code>dpiRules</code> parameter. For example, if you have two map expressions in the <code>dpiRules</code> parameter, the policy template would generate the body of the <for-each-rule> element once for each map expression.</p> <p>The <for-each-rule> element has a <code>ruleNumber</code> variable to sequentially track the processing of each map expression.</p> <p>You can use the test attribute to provide a condition for the rule; using this attribute would be the same as adding an <if> element.</p>



NOTE: When using special XML characters as part of the policy templates, they must be coded in XML. For example, the left angle bracket (`<`) must be coded as `<`.

The following example uses some elements to show a policy template that activates application-aware access list (AACL) services and service sets on an MX Series router by loading the configuration in text format using Junos XML management protocol.

```

<policy-templates>
  <target interface-class="MXEnterprise">
    <activation>
      <junoscript>
        <rpc>
          <load-configuration action="replace" format="text">
            <configuration-text>
services {
  aacl {
    rule AACL_[[[ interfaceName_ ]]] {
      match-direction input-output;
      <for-each-rule>
        term [[ ruleNumber ]]] {
          from {
            application junos:[[ app ]]];
          }
          then {
            <if test="fcl">
              forwarding-class [[ fcl ]]];
            </if>
            <if test="action~accept">
              count application;
            </if>
            [[ action ]]];
          }
        }
      </for-each-rule>
    }
  }
  service-set SSET_[[[ interfaceName_ ]]] {
    aacl-rules AACL_[[[ interfaceName_ ]]];
  }
}

```

```

        interface-service {
            service-interface ms-1/[[[ interfaceName~[^\.]+\d+(\d+/\d+/\d+.\d+)
]]];
        }
    }
}
interfaces {
    [[[ interfaceName~[^\.]+\d+ ]]] {
        unit [[[ interfaceName~[^\.]+\d+(\d+) ]]] {
            family inet {
                service {
                    input {
                        service-set SSET_[[[ interfaceName_ ]]]
                    }
                    output {
                        service-set SSET_[[[ interfaceName_ ]]]
                    }
                }
            }
        }
    }
}
}
</configuration-text>
</load-configuration>
</rpc>
</junoscript>
</activation>
</target>
</policy-templates>

```

If the example uses the following dpiRules substitution:

```

dpiRules=[{app="rtsp", action="accept", fcl="expedited-forwarding"},
          {app="bittorrent", action="discard"}]

```

The two map expressions in the dpiRules parameter might generate the following target configuration (with two terms) from the policy template example:

```

services {
    aacl {
        rule AAACL_xe_8_3_0_1001 {
            match-direction input-output;
            term 1 {
                from {
                    applications junos:rtsp;
                }
                then {
                    forwarding-class expedited-forwarding;
                    count application;
                    accept;
                }
            }
            term 2 {
                from {
                    applications junos:bittorrent;
                }
                then {

```



```

        discard;
    }
}
}
service-set SSET_xe_8_3_0_1001 {
    aacl-rules AACL_[[ interfaceName_ ]];
    interface-service {
        service-interface ms-1/3/0.1001;
    }
}
}
interfaces {
    xe-8/3/0 {
        unit 1001 {
            family inet {
                service {
                    input {
                        service-set SSET_xe_8_3_0_1001;
                    }
                    output {
                        service-set SSET_xe_8_3_0_1001;
                    }
                }
            }
        }
    }
}
}
}

```

Related Documentation

- [Substituting Parameters in Policy Templates on page 18](#)
- For information about the Junos XML management protocol, see *Junos XML Management Protocol Developer Guide*
- For information about service set configuration on routers running Junos OS, see *Junos OS Services Interfaces Library for Routing Devices*
- For information about accessing and configuring the third-party device using Telnet, see the device's software documentation

CHAPTER 4

Example

- Example: Using the DPI Script Service on page 31

Example: Using the DPI Script Service

To use the DPI script service provided:

1. Download the DPI script service to your system from the Juniper Networks website:
<https://www.juniper.net/support/downloads/?p=src#sw>

The files for supporting the DPI script service can be found in the SRC Demo and Sample Application software (*SDK+AppSupport+Demos+Samples.tar.gz* file).

The */SDK/scriptServices/dpiScriptService/lib/dpiss.jar* file contains the DPI script service implementation. Copy the *dpiss.jar* file to a location that is accessible to the SAE by a URL.

The */SDK/scriptServices/dpiScriptService/resource/dpiConfig.xml* file contains the sample configuration file that is included in the *dpiss.jar* file.

2. Import the sample data for the DPI script service using an LDAP browser.

The */SDK/scriptServices/dpiScriptService/ldif/dpiService.ldif* and */SDK/scriptServices/dpiScriptService/ldif/dpiSubscriber.ldif* files contain the sample service definition and subscriber configuration for setting up the script service.

To load the sample data into the database, you can use an LDAP tool, such as **ldapmodify**. To load data into the Juniper Networks database, you need the IP address of the database and the database credentials. The default bind distinguished name (DN) for the database is *cn=umcadmin, o=umc* and the password is *admin123*.

3. Modify the service substitutions for your device.

You can make these substitutions by defining the parameter substitutions in the DPI service with the SRC CLI or by passing the values through the enterprise portal.

4. Configure a subscription to the DPI service that is activated on login.

**Related
Documentation**

- For information about defining parameter substitutions with the SRC CLI, see [Creating the DPI Script Service \(SRC CLI\) on page 11](#) or [Configuring Subscriptions to the DPI Script Service on page 12](#)
- For information about passing the values through the SAE core API, see [Substituting Parameters in Policy Templates on page 18](#)
- *Subscriptions Overview*