

## Volume Usage Control with the SRC VTA



---

Published: 2012-04-05

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

Copyright © 2012, Juniper Networks, Inc. All rights reserved.

Juniper Networks, Junos, Steel-Belted Radius, NetScreen, and ScreenOS are registered trademarks of Juniper Networks, Inc. in the United States and other countries. The Juniper Networks Logo, the Junos logo, and JunosE are trademarks of Juniper Networks, Inc. All other trademarks, service marks, registered trademarks, 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.

Products made or sold by Juniper Networks 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.

*Volume Usage Control with the SRC VTA*  
Copyright © 2012, 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 <http://www.juniper.net/support/eula.html>. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

# Table of Contents

	About the Documentation . . . . .	xi
	Documentation and Release Notes . . . . .	xi
	Supported Platforms . . . . .	xi
	Documentation Conventions . . . . .	xi
	Documentation Conventions . . . . .	xii
	Documentation Feedback . . . . .	xiii
	Requesting Technical Support . . . . .	xiv
	Self-Help Online Tools and Resources . . . . .	xiv
	Opening a Case with JTAC . . . . .	xv
<b>Part 1</b>	<b>Overview</b>	
<b>Chapter 1</b>	<b>Software Features Overview . . . . .</b>	<b>3</b>
	SRC Component Overview . . . . .	3
<b>Chapter 2</b>	<b>Overview of Controlling Volume Usage with the SRC VTA . . . . .</b>	<b>9</b>
	Overview of the SRC VTA . . . . .	9
	Terminology . . . . .	9
	VTA Service and Subscriber Accounts . . . . .	10
	VTA Sessions . . . . .	11
	Volume-Based Services . . . . .	11
	SRC VTA Architecture and Connections to SRC Components . . . . .	11
	How the SRC VTA Works . . . . .	12
	Events . . . . .	13
	Event Handlers . . . . .	13
	Actions . . . . .	14
	Processors . . . . .	14
	Db-Engine Processor . . . . .	14
	SAE Proxy Processor . . . . .	15
	Mailer Processor . . . . .	15
	Scripts Processor . . . . .	16
	SRC VTA Operation . . . . .	16
	Overview of Configuring Event Handlers . . . . .	17
	Priority . . . . .	18
	Events . . . . .	18
	Event Attributes . . . . .	19
	Condition . . . . .	20
	Actions . . . . .	20

	Overview of Configuring Actions . . . . .	21
	On Error . . . . .	28
	Overview of Configuring the Event Queue . . . . .	28
	Setting the Size of the Event Queue . . . . .	28
	Calculating the Size of the Non-Persistent Event Queue . . . . .	29
	Overview of Managing VTA Accounts and Sessions . . . . .	30
	Identifying Subscribers, SAEs, and Sessions . . . . .	30
	Managing VTA Accounts and Sessions . . . . .	30
	Managing Subscriber Sessions and Service Sessions . . . . .	30
	Overview of Adjusting the Interim Accounting Interval . . . . .	31
	Using JavaScript Programs in VTA Configurations . . . . .	31
	Overview of the Web Application Server on C Series Controllers . . . . .	32
	Clustering . . . . .	33
	Local and Shared Configuration . . . . .	33
<b>Part 2</b>	<b>Configuration</b>	
<b>Chapter 3</b>	<b>Prerequisites for Running the SRC VTA . . . . .</b>	<b>37</b>
	Before You Configure the SRC VTA . . . . .	37
	Configuring the SAE to Send Tracking Events to the SRC VTA . . . . .	38
	Configuring the External Database . . . . .	40
	Configuring a Database to Store Account and Session Data (SRC CLI) . . . . .	41
	Installing the JDBC Driver .jar File . . . . .	42
	Configuring VTA Services and Policies . . . . .	42
	Configuring Subscribers and Subscriptions to VTA Services . . . . .	43
<b>Chapter 4</b>	<b>Configuration Tasks for Using NIC with the SRC VTA . . . . .</b>	<b>45</b>
	Locating the SAE That Manages a Subscriber for the SRC VTA . . . . .	45
	Configuring a NIC for the VTA (SRC CLI) . . . . .	45
	Configuring NIC Proxies for the VTA . . . . .	46
<b>Chapter 5</b>	<b>Configuration Tasks for the SRC Web Application Server . . . . .</b>	<b>49</b>
	Configuring the Web Application Server (SRC CLI) . . . . .	49
	Configuration Statements for the Web Application Server . . . . .	50
	Configuring Local Properties for the Web Application Server (SRC CLI) . . . . .	51
	Configuring the Web Application Server Shared Cluster Configuration (SRC CLI) . . . . .	52
	Configuring the Nodes in the Web Application Server Cluster (SRC CLI) . . . . .	53
	Configuring Remote Access to the Application Server (SRC CLI) . . . . .	54
	Configuring Access to the Application Server Through Secure HTTP . . . . .	54
	Configuring Access to the Application Server Through HTTP . . . . .	54
	Configuring Virtual Hosts for the Web Applications (SRC CLI) . . . . .	55
	Configuring User Accounts for Web Applications (SRC CLI) . . . . .	56
	Installing Web Applications in the Application Server . . . . .	57
	Starting the Web Application Server on a C Series Controller . . . . .	58

<b>Chapter 6</b>	<b>Configuration Tasks for the SRC VTA . . . . .</b>	<b>59</b>
	Configuring a VTA Shared Group Configuration (SRC CLI) . . . . .	59
	Creating and Configuring a VTA Shared Group Configuration (SRC CLI) . . . .	59
	Keys Used to Specify the Subscriber ID Solution (SRC CLI) . . . . .	60
	Keys Used to Specify the SAE Subscriber ID (SRC CLI) . . . . .	61
	Configuring the Connection Between the SRC VTA and the External Account and Session Database (SRC CLI) . . . . .	62
	Configuring Actions for a VTA Event Handler (SRC CLI) . . . . .	65
	Configuring Event Handlers (SRC CLI) . . . . .	66
	Configuring the Event Queue (SRC CLI) . . . . .	68
	Configuring the DB-Engine Processor for the SRC VTA Group (SRC-CLI) . . . . .	69
	Recording Balance Changes and Calculating the Average Usage Rate (SRC CLI) . . . . .	69
	Configuring the Initial Balance and Status of a Subscriber Account in the External Database (SRC CLI) . . . . .	70
	Configuring Scripts That Update Accounts (SRC CLI) . . . . .	71
	Configuring the Interim Account Interval and Usage Metric of a Service in the External Database (SRC CLI) . . . . .	72
	Variables Used to Define the Interim Accounting Interval for Services . . . . .	72
	Current Service Variables . . . . .	72
	Other Service Variables . . . . .	74
	Account Balance Variable . . . . .	74
	Sample Formulas for Interim Accounting Interval . . . . .	75
	Variables Used to Define the Usage Metric for Services . . . . .	75
	Sample Formulas for Usage Metrics for the SRC VTA . . . . .	76
	Configuring the Mailer Processor for the SRC VTA Group (SRC CLI) . . . . .	77
	Configuring the SRC VTA Mailer Processor to Send E-Mail Notifications (SRC CLI) . . . . .	77
	Configuring the SRC VTA to Send E-Mail Notifications (SRC CLI) . . . . .	78
	Configuring the SRC VTA Scripts Processor (SRC CLI) . . . . .	79
	Configuring the SRC VTA to Run Scripts . . . . .	80
	Configuring JavaScript Programs . . . . .	80
	Configuring External Scripts . . . . .	81
	Configuring VTA Logging (SRC CLI) . . . . .	83
	Logging Events Messages to a Text File . . . . .	83
	Logging Events Messages to a System Logging Server . . . . .	86
	Enabling, Disabling, and Restarting the VTA (SRC CLI) . . . . .	88
	Using One VTA Account for Multiple Subscriber Sessions . . . . .	89
<b>Chapter 7</b>	<b>Examples . . . . .</b>	<b>91</b>
	Example: Limiting Subscriber Access Based on Account Balances . . . . .	91
<b>Part 3</b>	<b>Administration</b>	
<b>Chapter 8</b>	<b>Managing the SRC VTA . . . . .</b>	<b>95</b>
	Deleting Balance Change History Records from the Database (SRC CLI) . . . . .	95
	Deleting Session History Records from the Database (SRC CLI) . . . . .	95
	Deleting Subscriber VTA Accounts (SRC CLI) . . . . .	96
	Modifying VTA Accounts and Service Sessions (SRC CLI) . . . . .	96

	Terminating Sessions (SRC CLI) .....	97
	Viewing a Subscriber's VTA Accounts (SRC CLI) .....	98
	Viewing Balance Change History for a Subscriber (SRC CLI) .....	98
	Viewing a Subscriber's Session History (SRC CLI) .....	98
	Viewing VTA Performance Statistics (SRC CLI) .....	99
	Overview of Testing the VTA Configuration .....	102
	Testing VTA Events (SRC CLI) .....	105
<b>Part 4</b>	<b>Troubleshooting</b>	
<b>Chapter 9</b>	<b>Troubleshooting Initial Configuration .....</b>	<b>109</b>
	Troubleshooting Database Deadlocks .....	109
<b>Part 5</b>	<b>Index</b>	
	Index .....	113

# List of Figures

Part 1	Overview	
Chapter 2	Overview of Controlling Volume Usage with the SRC VTA .....	9
	Figure 1: SRC VTA Architecture and Position in the SRC Network .....	12
	Figure 2: VTA Event Handler Model .....	14
	Figure 3: Operation of the SRC VTA .....	16





# List of Tables

	<b>About the Documentation . . . . .</b>	<b>xi</b>
	Table 1: Notice Icons . . . . .	xii
	Table 2: Notice Icons . . . . .	xii
	Table 3: Text Conventions . . . . .	xii
<b>Part 1</b>	<b>Overview</b>	
<b>Chapter 1</b>	<b>Software Features Overview . . . . .</b>	<b>3</b>
	Table 4: Descriptions of SRC Components . . . . .	3
<b>Chapter 2</b>	<b>Overview of Controlling Volume Usage with the SRC VTA . . . . .</b>	<b>9</b>
	Table 5: SRC VTA Terms . . . . .	10
	Table 6: VTA Event Types . . . . .	18
	Table 7: Event Attributes . . . . .	19
	Table 8: VTA Functions and Input Parameters . . . . .	22
<b>Part 2</b>	<b>Configuration</b>	
<b>Chapter 3</b>	<b>Prerequisites for Running the SRC VTA . . . . .</b>	<b>37</b>
	Table 9: Settings for Filter Strings . . . . .	39
<b>Chapter 6</b>	<b>Configuration Tasks for the SRC VTA . . . . .</b>	<b>59</b>
	Table 10: Keys That the SRC VTA Constructs to Manage Accounts and Sessions . . . . .	61
	Table 11: Keys That the SRC VTA Constructs to Manage Subscriber and Service Sessions . . . . .	61
	Table 12: Examples of Interim Accounting Interval . . . . .	75
	Table 13: Examples of Formulas That Calculate Use of Network Resources . . . . .	76
	Table 14: Named Severity Levels . . . . .	84
	Table 15: Examples of Filters for Event Messages . . . . .	85
	Table 16: Named Severity Levels . . . . .	87
	Table 17: Examples of Filters for Event Messages . . . . .	88
<b>Part 3</b>	<b>Administration</b>	
<b>Chapter 8</b>	<b>Managing the SRC VTA . . . . .</b>	<b>95</b>
	Table 18: Arguments Used to Modify Accounts . . . . .	97
	Table 19: VTA Event Types and Test Event Configuration Statements . . . . .	102
	Table 20: Attributes for Subscriber- and Service-Tracking Test Events . . . . .	103



# About the Documentation

- Documentation and Release Notes on page xi
- Supported Platforms on page xi
- Documentation Conventions on page xi
- Documentation Feedback on page xiii
- Requesting Technical Support on page xiv

## 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 <http://www.juniper.net/techpubs/>.

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 <http://www.juniper.net/books>.

## Supported Platforms

---

For the features described in this document, the following platforms are supported:

- C Series

## Documentation Conventions

---

Table 1 on page xii 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.

## Documentation Conventions

Table 1 on page xii defines the notice icons used in this guide. Table 3 on page xii 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.

Table 3: Text Conventions

Convention	Description	Examples
<b>Bold text like this</b>	<ul style="list-style-type: none"> <li>Represents keywords, scripts, and tools in text.</li> <li>Represents a GUI element that the user selects, clicks, checks, or clears.</li> </ul>	<ul style="list-style-type: none"> <li>Specify the keyword <b>exp-msg</b>.</li> <li>Run the <b>install.sh</b> script.</li> <li>Use the <b>pkgadd</b> tool.</li> <li>To cancel the configuration, click <b>Cancel</b>.</li> </ul>

Table 3: Text Conventions (*continued*)

<b>Bold text like this</b>	Represents text that the user must type.	<b>user@host# set cache-entry-age cache-entry-age</b>
<b>Fixed-width text like this</b>	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>
<b>Regular sans serif typeface</b>	<ul style="list-style-type: none"> <li>Represents configuration statements.</li> <li>Indicates SRC CLI commands and options in text.</li> <li>Represents examples in procedures.</li> <li>Represents URLs.</li> </ul>	<ul style="list-style-type: none"> <li><b>system ldap server{ stand-alone;</b></li> <li>Use the <b>request sae modify device failover</b> command with the <b>force</b> option</li> <li><b>user@host# ...</b></li> <li><b><a href="http://www.juniper.net/techpubs/software/management/sdx/api-index.html">http://www.juniper.net/techpubs/software/management/sdx/api-index.html</a></b></li> </ul>
<i>Italic sans serif typeface</i>	Represents variables in SRC CLI commands.	<b>user@host# set local-address local-address</b>
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"> <li>Emphasizes words.</li> <li>Identifies book names.</li> <li>Identifies distinguished names.</li> <li>Identifies files, directories, and paths in text but not in command examples.</li> </ul>	<ul style="list-style-type: none"> <li>There are two levels of access: <i>user</i> and <i>privileged</i>.</li> <li><i>SRC-PE Getting Started Guide</i>.</li> <li><i>o=Users, o=UMC</i></li> <li>The <i>/etc/default.properties</i> file.</li> </ul>
Backslash	At the end of a line, indicates that the text wraps to the next line.	<pre> Plugin.radiusAcct-1.class=\ net.juniper.smgmt.sae.plugin\ RadiusTrackingPluginEvent </pre>
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

## Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can send your comments to [techpubs-comments@juniper.net](mailto:techpubs-comments@juniper.net), or fill out the documentation feedback form at

<https://www.juniper.net/cgi-bin/docbugreport/> . If you are using e-mail, be sure to include the following information with your comments:

- Document or topic name
- URL or page number
- Software release 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 J-Care or JNASC 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 <http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf> .
- Product warranties—For product warranty information, visit <http://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: <http://www.juniper.net/customers/support/>
- Search for known bugs: <http://www2.juniper.net/kb/>
- Find product documentation: <http://www.juniper.net/techpubs/>
- Find solutions and answer questions using our Knowledge Base: <http://kb.juniper.net/>
- Download the latest versions of software and review release notes: <http://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications: <https://www.juniper.net/alerts/>
- Join and participate in the Juniper Networks Community Forum: <http://www.juniper.net/company/communities/>
- Open a case online in the CSC Case Management tool: <http://www.juniper.net/cm/>

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

## Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at <http://www.juniper.net/cm/> .
- 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 <http://www.juniper.net/support/requesting-support.html> .





## PART 1

# Overview

- [Software Features Overview on page 3](#)
- [Overview of Controlling Volume Usage with the SRC VTA on page 9](#)



## 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
<b>Server Components</b>	
Service activation engine (SAE)	<ul style="list-style-type: none"><li>• 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.</li><li>• Collects accounting information about subscribers and services from routers, and stores the information in RADIUS accounting servers, flat files, and other accounting databases.</li><li>• 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.</li></ul>
Subscriber Information Collector (SIC)	Used in conjunction with the MX Series router running the packet-triggered subscribers and policy control (PTSP) solution, the SIC listens for RADIUS accounting events from IP edge devices (accounting clients) and stores them in the Session State Registrar (SSR), or 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.
Juniper Policy Server (JPS)	Acts as a policy decision point (PDP) and policy enforcement point (PEP) that manages the relationships between application managers and CMTS devices in a PCMM environment.

Table 4: Descriptions of SRC Components *(continued)*

Component	Description
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.
<b>Repository</b>	
Directory	Provides a repository of subscriber information, services, policies, and service portal configurations. The SRC software uses the Lightweight Directory Access Protocol (LDAP) for interactions with the directory.
Juniper Networks Database	Repository for SRC data on a C Series Controller. When used in conjunction with the
Session State Registrar (SSR)	The SSR is a stateless, highly reliable and highly available database cluster. When used in conjunction with an MX Series router running the packet-triggered subscribers and policy control (PTSP) solution, the SSR stores the IP edge attachment subscriber sessions data learned from IP edge devices in the centralized SSR database.
Volume Tracking Application	<p>The SRC Volume Tracking Application (SRC VTA) 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>
<b>SRC Configuration and Management Tools</b>	
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.
<b>Service Management Applications (Run on external system)</b>	

Table 4: Descriptions of SRC Components (*continued*)

Component	Description
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.
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

Table 4: Descriptions of SRC Components *(continued)*

Component	Description
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.
SRC Volume Tracking Application (SRC VTA)	Monitors subscriber resource usage to allow service providers to offer flexible usage quotas, limit bandwidth to subscribers that overuse network resources, and to notify subscribers who may have been compromised by viruses or worms that overuse network resources.
<b>Demonstration Applications (available on the Juniper Networks Web site)</b>	
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>
Intrusion detection and protection (IDP) integration applications	Integrates IDP into an SRC-managed environment to manage malicious traffic sent to or received by subscribers. The IDP integration applications run on a Solaris platform.
Instant Virtual Extranet (IVE) Host Checker integration application	Integrates the IVE Host Checker into an SRC-managed environment to verify that the subscriber systems used to connect to a service provider comply with the service provider's policies. The IVE Host Checker integration application runs on a Solaris platform.
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.
Prepaid Account Administration application	Manages prepaid accounts for the prepaid services demonstration application. The Prepaid Account Administration application runs on a Solaris platform.

Table 4: Descriptions of SRC Components *(continued)*

Component	Description
Prepaid service application	Demonstrates how the SRC software might be used to manage prepaid accounts. The Prepaid service 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.
Traffic-Mirroring Administration application	Manages and monitors mirroring tasks. The Traffic-Mirroring Administration application runs on a Solaris platform.
Traffic-Mirroring Application	Mirrors subscriber traffic on any subscriber access platform supported by the SRC software. Provides the Traffic-Mirroring Administration portal to manage the mirroring of subscriber traffic. The Traffic-Mirroring application runs on a Solaris platform.
<b>Auxiliary Applications</b>	
Application server	Enables J2EE applications, including Web applications, to be used with the SRC software. These third-party applications run on a system other than a C Series Controller.
Other applications	Third-party applications created to run in an SRC environment.

**Related Documentation**

- SRC Product Description





## CHAPTER 2

# Overview of Controlling Volume Usage with the SRC VTA

- [Overview of the SRC VTA on page 9](#)
- [SRC VTA Architecture and Connections to SRC Components on page 11](#)
- [How the SRC VTA Works on page 12](#)
- [SRC VTA Operation on page 16](#)
- [Overview of Configuring Event Handlers on page 17](#)
- [Overview of Configuring Actions on page 21](#)
- [Overview of Configuring the Event Queue on page 28](#)
- [Overview of Managing VTA Accounts and Sessions on page 30](#)
- [Overview of Adjusting the Interim Accounting Interval on page 31](#)
- [Using JavaScript Programs in VTA Configurations on page 31](#)
- [Overview of the Web Application Server on C Series Controllers on page 32](#)

## Overview of the SRC VTA

---

The SRC Volume-Tracking Application (SRC VTA) 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.

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. You can configure multiple VTAs.

If you use the SRC VTA with the deep packet inspection (DPI) feature, you can control the volume of traffic for specific applications, such as peer-to-peer file sharing.

## Terminology

[Table 5 on page 10](#) defines terms that are used in the SRC VTA documentation and sample data.

For information on loading the sample data for the VTA see, Loading Sample Data into a Juniper Networks Database (SRC CLI).

**Table 5: SRC VTA Terms**

Term	Definition
Bought quota	Allowance of data volume that subscribers purchase and can transfer (upload or download) at any time. (This term is used in sample and typical VTA configurations and is not inherent in the SRC VTA itself.)
Bought account	Record that details a subscriber's use of bought quota. (This term is used in sample and typical VTA configurations and is not inherent in the SRC VTA itself.)
Periodic quota	Allowance of data volume that a service provider allocates to subscribers on a recurrent basis. Subscribers use this allowance to upload or download data. (This term is used in sample and typical VTA configurations and is not inherent in the SRC VTA itself.)
Periodic account	Record that tracks a subscriber's use of periodic quota. (This term is used in sample and typical VTA configurations and is not inherent in the SRC VTA itself.)
Quota service	Service for which a VTA monitors usage. The SRC VTA activates the service for subscribers when they have a positive balance in their VTA accounts, and deactivates the service when the VTA account has a negative balance. (This term is used in sample and typical VTA configurations and is not inherent in the SRC VTA itself.)
VTA account	Record of credit and debit entries that track a subscriber's use of a particular network resource.
VTA session	Period of activity between a VTA subscriber and a VTA.

## VTA Service and Subscriber Accounts

A VTA account represents the resources available to a service or a subscriber. You can configure VTA accounts and then charge a particular service or subscriber's usage against the account. Each subscriber or service can have a different quota, or allowance of data volume.

You can set up the way the VTA charges accounts and how account balances are updated.

You can also configure actions in response to changes in account balances. Available actions include stopping a service, starting a service, updating an account balance, sending an e-mail, and running a script. For example, if account A is emptied, the action might be to stop services X and Y, and start service Z.

The SRC VTA requires a relational database to store information about accounts. The SRC VTA installation includes sample schemas for the MySQL and Oracle databases.

## VTA Sessions

The SRC VTA tracks subscriber activity through VTA sessions. A VTA session does not necessarily correspond to an individual subscriber session or service session. For example, a single service session can correspond to multiple VTA sessions if the service session covers multiple billing periods.

The SRC VTA can track more than just the volume and time of a service session, it can track any state of a subscriber derived from SAE plug-in events and respond to the state change.

The SRC VTA requires a relational database to store information about sessions. The SRC VTA installation includes sample schemas for the MySQL and Oracle databases.

## Volume-Based Services

The SRC VTA lets you set triggers at multiple levels to provide flexible and extensive volume-based services. For example:

- When the volume remaining for the account is 300 MB, turn on the internet-256 service, turn off the internet-512 service, and send an e-mail to the subscriber.
- When the volume level reaches 100 MB, send an e-mail warning to the subscriber.
- When the volume level is 0 MB, turn on the continue-TCP-only service, turn off the internet-256 service, send an e-mail to the subscriber, and notify the accounting server.
- When the volume level is -100 MB, turn off the continue-TCP-only service, send an e-mail to subscriber, and notify the accounting server.

### Related Documentation

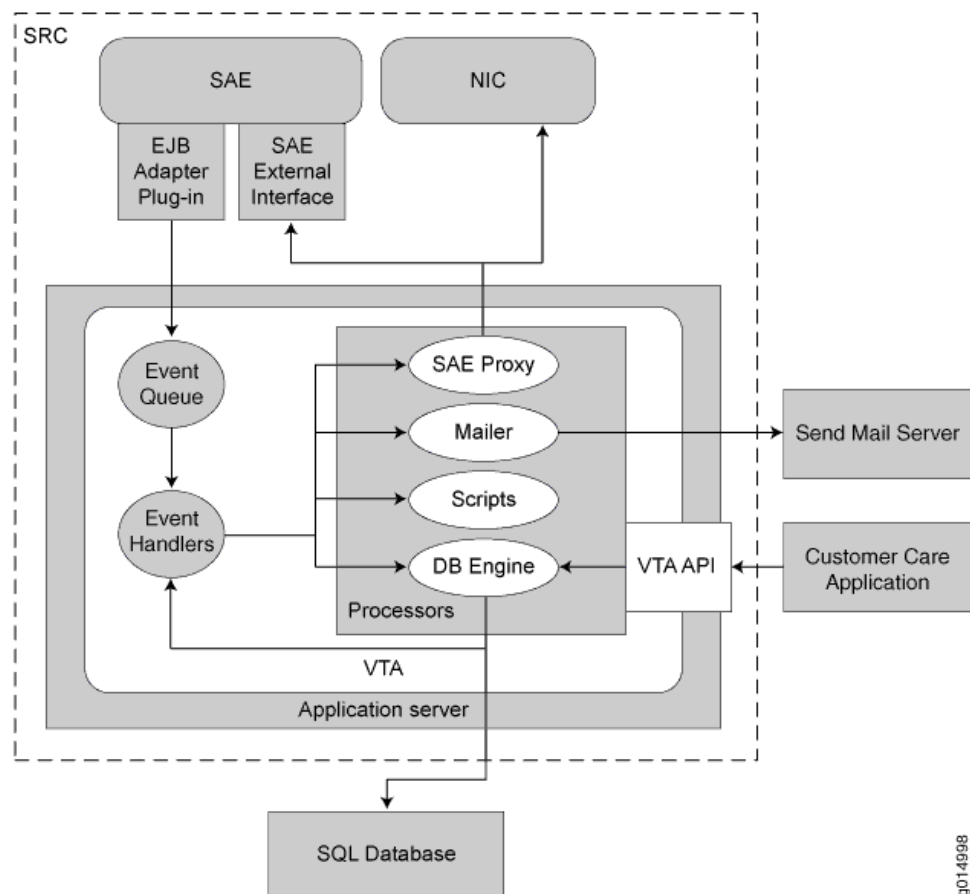
- [How the SRC VTA Works on page 12](#)
- [SRC VTA Operation on page 16](#)
- [Before You Configure the SRC VTA on page 37](#)
- [SRC VTA Architecture and Connections to SRC Components on page 11](#)

---

## SRC VTA Architecture and Connections to SRC Components

[Figure 1 on page 12](#) shows the SRC VTA architecture and the position of a VTA in the SRC network.

Figure 1: SRC VTA Architecture and Position in the SRC Network



- Related Documentation**
- [Overview of the SRC VTA on page 9](#)
  - [How the SRC VTA Works on page 12](#)
  - [SRC VTA Operation on page 16](#)

## How the SRC VTA Works

The SRC VTA manages subscriber accounts using a rule-driven event-processing system that can prioritize the actions taken for certain conditions. The SRC VTA is triggered by events, such as the logging in of subscribers, the use of network services, or the changing of account balances. These events can cause actions, such as updating account balances, starting or stopping network services, or running scripts to perform external actions.

A VTA processes external events based on its configuration. A VTA configuration is made up of:

- Event handlers
- Actions
- Processors

## Events

Each VTA event corresponds to one subscriber and contains some attributes. The SRC VTA supports the following types of events:

- Service and subscriber-tracking events from the SAE; for example, start or stop tracking events.
- Account update events triggered by updating database accounts.
- Callback events triggered by a VTA API call.

## Event Handlers

An event handler defines how the SRC VTA processes an event. VTA event handlers consist of:

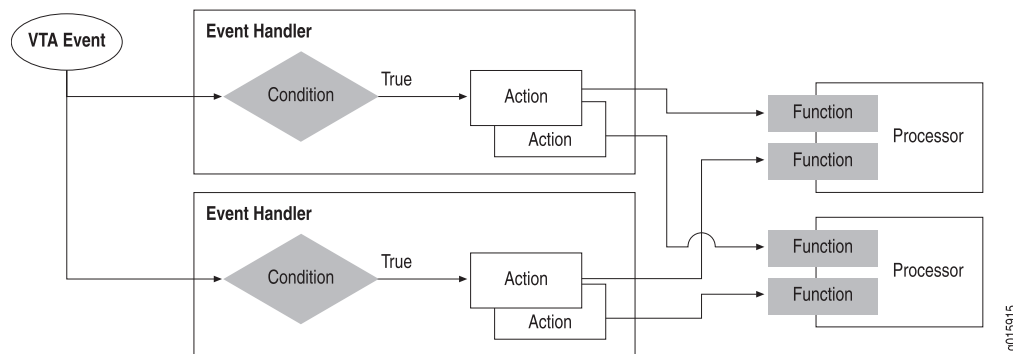
- Type of event—Tracking, update, or callback event; for example, a subscriber start-tracking event or a service start-tracking event.
- Priority—Priority at which event handlers are evaluated and executed. Event handlers are evaluated and executed from high to low priority.
- Condition—Condition that the event handler evaluates to determine whether the event handler should handle the event.
- Actions—List of actions to be performed by the event handler.

You can set up multiple event handlers to process events. For example, the first event handler could retrieve the balance for a quota account, and the next event handler could refill the quota account, depending on whether the condition of the second event handler is met.

[Figure 2 on page 14](#) shows the event handler model. The SRC VTA processes an event as follows:

1. The event handler with the highest priority receives the event, determines whether the event's type is the same as the event type of the event handler, and determines whether the event satisfies the condition of the event handler.
2. If the condition is met, the SRC VTA performs the corresponding actions based on the event attributes. An action invokes a function and provides the parameters required by that function to the processor.
3. When an event handler finishes processing an event, the next applicable event handler according to the priority of the event handler processes the event.

Figure 2: VTA Event Handler Model



## Actions

You specify actions that the SRC VTA takes in response to events; for example, updating an account balance, starting a service, or stopping a service. An action is modeled as a call to a function. If an event matches the type and condition requirements of the event handler, then all actions defined for the event handler process the event, one after another, in the order the actions are configured.

An action can update event attributes or add new attributes to an event for subsequent processing of the same event by another action in the same event handler, or by actions in subsequent event handlers. The updated attributes can also be used to change whether the event satisfies a subsequent event handler's condition.

An action configuration includes the following:

- Function—Function that the action invokes
- Parameter—Parameters and corresponding values to be passed to the function

## Processors

Processors implement the functions that receive and process events. The SRC VTA has four processors:

- Db-engine processor—Acts as a proxy to a database
- Mailer processor—Sends e-mail notifications when certain events occur
- SAE proxy processor—Acts as a proxy to the SAE. (Requires no configuration)
- Scripts processor—Runs external scripts or JavaScript programs

## Db-Engine Processor

The db-engine processor acts as a proxy to the external database. You can use the functions provided by the db-engine processor to:

- Calculate the use of network resources for a service.
- Calculate the interim accounting interval for each service based on a subscriber's remaining resources and use of the service.
- Update VTA accounts with a JavaScript program.
- Terminate a VTA session. This feature is usually used at the end of a billing period so that you can finish collecting data for the current billing period and start a new VTA session for the new billing period.

## SAE Proxy Processor

The SAE proxy processor is a proxy to the SAE external interface that resolves the subscriber interface based on the event types to which functions are applied.

- If a function is applied to SAE subscriber-tracking or service-tracking events, the processor finds the SAE reference in the event message.

There is an exception if the **current-subscriber-only** parameter is set to false. In this case, the function finds subscribers in all SAEs with the NIC.

- If a function is applied to other events, the processor uses the subscriber's ID in the event as the key for the NIC to find the SAE reference. The VTA uses the SAE reference and the **subscriber-id-solution** option that you specify under the VTA group to look up the SAE and the subscriber.

You can use the functions provided by the SAE proxy processor to:

- Set an interim interval for a service.
- Set a service session timeout for a subscription.
- Set a session timeout for a subscriber.
- Start a subscription to a service. You can specify the parameter substitutions to use when the service is started.
- Stop a subscription to a service. You can include a reason for stopping the subscription. When the service is stopped, the reason is sent to the billing system so it can differentiate between service stops.

The SAE proxy processor does not require configuration. To use the functions provided by the SAE proxy processor, you configure an action that calls the respective function for the event handler.

## Mailer Processor

You can use the functions provided by the mailer processor to set up the SRC VTA to send e-mail notifications when certain events occur. You can specify that e-mail notifications be sent to subscribers, system administrators, or an automated billing system.

## Scripts Processor

The scripts processor can invoke external executable scripts or JavaScripts. We recommend using JavaScript, where possible, for better performance.

- External scripts are executable programs, such as shell scripts, that are available on the SRC VTA's host. Each external script can perform a task and return a value. If the script returns a value, the value can be added to the current event as an event attribute.
- JavaScript programs are used to process attributes of a VTA event and can also be used for any arbitrary purpose, just like external scripts. For example, a JavaScript program can convert a VTA event attribute in a timestamp to a date string and add it to the event as a new attribute. The attribute can then be used for subsequent actions, such as sending an e-mail notification to the subscriber. The JavaScript program can refer to any attributes of the event being processed, and it must return a value.

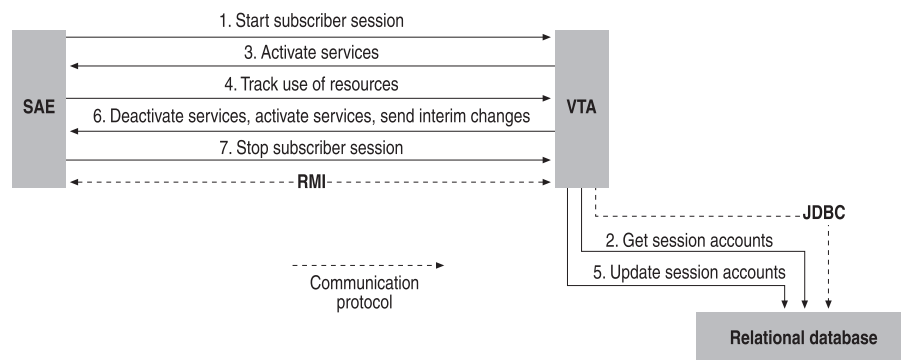
### Related Documentation

- [Overview of the SRC VTA on page 9](#)
- [SRC VTA Operation on page 16](#)
- [SRC VTA Architecture and Connections to SRC Components on page 11](#)
- [Overview of Configuring Event Handlers on page 17](#)
- [Overview of Configuring Actions on page 21](#)

## SRC VTA Operation

Figure 3 on page 16 illustrates the SRC VTA operation process.

Figure 3: Operation of the SRC VTA



The SRC VTA operates as follows:



1. When an event that activates a service occurs (for example, a subscriber logs in), the SAE sends a session start event to the SRC VTA through the ejb-adapter plug-in you configure on the SAE.
2. Optionally, the SRC VTA queries a database for the subscriber's current set of sessions and accounts.
3. Depending on the configuration of the SRC VTA, it may activate or deactivate services based on the subscriber's use of resources.
4. The SRC VTA tracks the subscriber's use of resources for a period of time.
5. The SRC VTA updates sessions and account balances in the database.
6. The SRC VTA sends to the SAE changes in the interim accounting interval and takes action to limit excessive bandwidth use or to allow increased bandwidth use.
7. When an event that deactivates a service occurs (for example, the subscriber logs out), the SRC VTA updates the VTA session, closes the VTA session, and may update the account balances.

Events received through the VTA API can also cause the SRC VTA to activate services, deactivate services, or change the accounting interval.

**Related  
Documentation**

- [Overview of the SRC VTA on page 9](#)
- [How the SRC VTA Works on page 12](#)
- [SRC VTA Architecture and Connections to SRC Components on page 11](#)

---

## Overview of Configuring Event Handlers

---

Event handlers define how the SRC VTA processes events. A VTA event handler configuration includes the following options:

- **Events**—Type of event including tracking, update, or callback event; for example, a service-start tracking event.
- **Priority**—Priority at which event handlers are evaluated and executed. Event handlers are evaluated and executed from low to high.
- **Condition**—Condition that the event handler evaluates to determine whether the event handler should process the event.
- **Actions**—List of actions to be performed by the event handler.

You can set up multiple event handlers to process events. For example, the first event handler could retrieve the balance for a quota account, and the next event handler could refill the quota account depending on whether the condition of the second event handler is met.

When an event is received, the corresponding event type and condition configured for the event handler are evaluated based on the priority. When a condition is met, the corresponding actions are performed according to the event attributes. The action can update or add event attributes to the events for subsequent processing of the same

event. An action can invoke a function provided by any processor. After an event handler has completed its processing, event processing continues with the next applicable event handler.

When an event is sent by the ejb-adaptor plug-in on the SAE, it is received by the `SAEEventListener`, which places the event in the event queue. Because you can have multiple VTA instances, each VTA has a separate `SAEEventListener`. The VTA takes events one by one and presents them to an ordered sequence of event handlers. Each event handler includes a list of actions. If the event handler is configured to handle the event, it acts on the event based on the configured actions—for example, updating a balance or starting a service. Each action can specify a function.

## Priority

When you configure an event handler, you need to specify the priority for evaluating and running the event handler. The priority is an integer where the smaller number has the higher priority. The event handler with the highest priority receives the event, determines whether the event's type is the same as the event type of the event handler, and determines whether the event satisfies the condition of the event handler. When an event handler finishes processing an event, the next applicable event handler according to the priority of the event handler processes the event.

## Events

Each VTA event corresponds to one subscriber and contains some attributes. The SRC VTA supports the following event types:

- Service and subscriber tracking events from the SAE; for example, start or stop tracking events.
- Account update events triggered by updating database accounts.
- Callback events triggered by a VTA API call.

Various events can be received by the VTA from the SAE. [Table 6 on page 18](#) describes the events supported by the SRC VTA.

**Table 6: VTA Event Types**

Event	Description
account-update	Database update event
callback:callid	External callback event for the specified call
service-interim:service name	Service interim—tracking event for the specified service
service-start:service name	Service start—tracking event for the specified service
service-stop:service name	Service stop—tracking event for the specified service
user-interim	User interim—tracking event

Table 6: VTA Event Types (*continued*)

user-start	User start—tracking event
user-stop	User stop—tracking event

The *service-name* is the name of any configured SRC service.

To process an event, the event handler must be configured to handle the particular event type. You configure which events are handled by the event handler with the **shared vta group name event-handler event-handler-name events events** statement. Separate events with a comma. You can configure an event handler to handle multiple event types. If an event handler is configured to handle an event, it can pass that event to a sequence of actions.

An example of an event is:

**service-start: *QuotaInternet*, callback: *TerminateSession***

### Event Attributes

Each event carries attributes. [Table 7 on page 19](#) describes the types of attributes that are available for each type of event.

Table 7: Event Attributes

Event Type	Available Attributes
Service and subscriber tracking events from the SAE	<ul style="list-style-type: none"> <li>Plug-in attributes, such as PA_SERVICE_NAME or PA_LOGIN_NAME, associated with an SAE plug-in event. For a list of SAE plug-in events, see <a href="#">"Overview of Testing the VTA Configuration" on page 102</a>.</li> <li>currentTime attribute—Time since January 1, 1970 UTC when the SRC VTA begins passing the event to the event handlers (events are queued in the event queue and then passed to the event handlers). The value is the number of milliseconds in the range 0–9223372036854775807.</li> <li>subscriberId—Subscriber ID based on attributes of a service or a subscriber-tracking event. The subscriberId event attribute is a result of the calculation. It identifies the subscriber of the corresponding VTA event.</li> </ul>
Account update events	<ul style="list-style-type: none"> <li>old_status_accountname—The old status of the account.</li> <li>new_status_accountname—Returns the new status of the specified account.</li> <li>old_lastUpdateTime_accountname—Returns the old last update time of the account.</li> <li>new_lastUpdateTime_accountname—Returns the new last update time of the account.</li> <li>old_balance_accountname—Returns the old balance of the account.</li> <li>new_balance_accountname—Returns the new balance of the specified account.</li> <li>currentTime—Time since January 1, 1970 UTC, when the SRC VTA begins passing the event to the event handlers (events are queued in the event queue and then passed to the event handlers). The value is the number of milliseconds in the range 0–9223372036854775807.</li> <li>subscriberId—Subscriber ID based on attributes of a service or a subscriber-tracking event. The subscriberId event attribute is a result of the calculation. It identifies the subscriber of the corresponding VTA event.</li> </ul>

Table 7: Event Attributes (*continued*)

Event Type	Available Attributes
Callback events	<ul style="list-style-type: none"> <li>callId—When the callback event type is invoked the <i>callId</i> value is specified. The <i>callId</i> value is placed in to the event using this event attribute and then processed. If an event handler is configured to handle this event type, the actions specified for the event handler can invoke functions, which have access to the <i>callId</i> in this event attribute.</li> <li>currentTime—Time since January 1, 1970 UTC, when the SRC VTA begins passing the event to the event handlers (events are queued in the event queue and then passed to the event handlers). The value is the number of milliseconds in the range 0–9223372036854775807.</li> <li>subscriberId—Subscriber ID based on attributes of a service or a subscriber-tracking event. The subscriberId event attribute is a result of the calculation. It identifies the subscriber of the corresponding VTA event.</li> </ul>

## Condition

Each event handler evaluates conditions to determine whether it should handle the event. You specify the condition as a script written in the JavaScript programming language that must return one of the following values:

- True—Event handler should handle the event.
- False—Event handler should not handle the event.

If the condition is met, the VTA performs the corresponding actions based on the event attributes. An action invokes a function and provides the parameters required by that function to the db-engine processor. If no condition is specified, true is returned as the value. If a referenced attribute does not exist in the event, the referenced attribute's value is null. Following is an example condition:

```
var newBalance=<balance_BoughtQuota>+<balance_PeriodicQuota>;
if(<old_balance_PeriodicQuota>==null)
  <old_balance_PeriodicQuota>=<balance_PeriodicQuota>;
if(<old_balance_BoughtQuota>==null)
  <old_balance_BoughtQuota>=<balance_BoughtQuota>;
return <old_balance_PeriodicQuota>+<old_balance_BoughtQuota><=0&&newBalance>0;
```



**NOTE:** In any condition or other JavaScript script, event attributes are referred to by enclosing them in angle < > brackets.

## Actions

When you configure an event handler, you specify actions that the event handler executes in response to an event; for example, updating an account balance, starting a service, or stopping a service. An action is performed only if the event matches the specified event type and condition you configure for the event handler. An action can invoke functions provided by any processor.



**NOTE:** We recommend that when you configure event handlers and their actions, you ensure that for any given event, all database operations are performed before any other operations that have permanent effects. This is because if a database error occurs—for example, due to normal contention for database records between different event threads—the VTA rolls back the current database transaction (no changes are made to the database) and then restarts processing the event. If the event performs some other operation other than database operations before such an error, such as start a service, then that other operation is performed again when the event is reprocessed following the error.

**Related  
Documentation**

- [Overview of Testing the VTA Configuration on page 102](#)
- [How the SRC VTA Works on page 12](#)
- [Configuring Event Handlers \(SRC CLI\) on page 66](#)
- [Overview of Configuring Actions on page 21](#)
- [Configuring Actions for a VTA Event Handler \(SRC CLI\) on page 65](#)

---

## Overview of Configuring Actions

Actions are executed by event handlers in response to an event. For example, the action may update an account balance, start a service, or stop a service. You configure what action the event handler takes in response to an event when you configure an event handler. An action is performed only if an event matches the event type and condition specified in the event handler configuration. An action can invoke functions provided by any processor.

An action can update event attributes or add new attributes to an event for subsequent processing of the same event by subsequent actions in the same event handler, or in subsequent event handlers.

You define an action with the `edit shared vta group name action action-name` configuration statement. You refer to a previously defined action when you configure an event handler. Event handlers can refer to multiple actions. To configure an event handler to use a particular action, specify it as follows:

```
[edit shared vta group name event-handler name]  
user@host# set actions actions
```



**NOTE:** We recommend that when you configure event handlers and their actions, you ensure that for any given event, all database operations are performed before any other operations that have permanent effects. This is because if a database error occurs—for example, due to normal contention for database records between different event threads—the VTA rolls back the current database transaction (no changes are made to the database) and then restarts processing the event. If the event performs some other operation other than database operations before such an error, such as start a service, then that other operation is performed again when the event is reprocessed following the error.

When you configure an action, you specify the following options:

- **Function**—The function that the action invokes
- **Parameter**—The parameters and corresponding values to be passed to the function
- **On-error**—What the event handler does in response to an error

Functions are variables that an action invokes, for example, to update database accounts. The function takes as input the configured parameters, the event, and the event context (which may include values created by previously executed actions). The SRC VTA provides a set of available functions. Each function takes a different set of input parameters. [Table 8 on page 22](#) describes the available VTA functions and their associated input parameters.

**Table 8: VTA Functions and Input Parameters**

Function Name	Description	Input Parameters
db-engine-calculate-interim	<p>Calculates the interim interval in the service-tracking event by using the interim interval function for the service, as defined with the <b>shared vta group name processor db-engine service name interim-interval-function</b> configuration statement. This function adds the following attribute to the event after the function is executed:</p> <ul style="list-style-type: none"> <li>• <i>interimInterval</i>—Interim interval of the service</li> </ul>	None

Table 8: VTA Functions and Input Parameters (*continued*)

db-engine-calculate-usage	<p>Calculates usage in the service-tracking event by using the usage metric function for the service, as defined with the <b>shared vta group name processor db-engine service name usage-metric-function</b> configuration statement. This function adds the following attributes to the event after the function is executed:</p> <ul style="list-style-type: none"> <li>• <i>currentUsage</i>—Usage since the previous usage report</li> <li>• <i>sessionSinceLastReport</i>—Session length since the previous usage report</li> </ul>	None
db-engine-get-accounts	<p>Retrieves account data for the corresponding subscriber for the event. Data for multiple accounts is retrieved; accounts are identified by the <i>accountName</i> suffix. Accounts are defined with the <b>shared vta group name processor db-engine account name</b> configuration statement, where the account name corresponds to the <i>_accountName</i> suffix. Subsequent event handlers of the event can use the retrieved data. This function adds the following attributes to the event after the function is executed:</p> <ul style="list-style-type: none"> <li>• <i>balance_accountName</i>—Balance for the account.</li> <li>• <i>lastUpdateTime_accountName</i>—Last update time in milliseconds since January 1, 1970 UTC for the account.</li> <li>• <i>status_accountName</i>—Status of the account.</li> </ul>	None
db-engine-terminate-session	<p>Closes active VTA sessions (for the subscriber for which the event occurred) that have a status of Start or Interim. It does not stop the corresponding services in the SAE. You can use this function to stop a session at the end of a billing period. Usage data collected after the VTA session is stopped is stored in new VTA session records.</p>	None

Table 8: VTA Functions and Input Parameters (*continued*)

db-engine-update-accounts	<p>Runs an account update script that changes the account balances of the corresponding subscriber for the event. Scripts are defined with the <b>shared vta group name processor db-engine account-update-script name</b> configuration statement. The predefined script name is used as input to this function. This function adds the following attributes to the event after the function is executed:</p> <ul style="list-style-type: none"> <li>• <i>balance_accountName</i>—Balance for the account after it is updated.</li> <li>• <i>lastUpdateTime_accountName</i>—Last update time in milliseconds since January 1, 1970 UTC for the account after it is updated.</li> <li>• <i>status_accountName</i>—Status of the account after it is updated.</li> </ul>	<p><b>script-name</b>—Name of the account update script defined in the db-engine processor.</p>
mailer-send	<p>Sends e-mail notifications when certain events occur. You can specify that e-mail notifications be sent to anyone, for example, subscribers, system administrators, or an automated billing system.</p>	<ul style="list-style-type: none"> <li>• <b>recipient</b>—Destination e-mail address</li> <li>• <b>from</b>—Source e-mail address</li> <li>• <b>subject</b>—Subject of the e-mail</li> <li>• <b>text</b>—Content of the e-mail</li> </ul>
sae-set-interim-interval	<p>Sets the interim accounting interval for the service session identified in the event to be the value of the <i>interimInterval</i> attribute currently found in the event. Before this function is called, the db-engine-calculate-interim function of the db-engine processor must be called in order to place the <i>interimInterval</i> attribute into the event.</p>	<p><b>current-subscriber-only</b>—Specifies whether the function is applied only to the subscriber identified in the event or to all subscribers who have the same subscriber ID.</p> <ul style="list-style-type: none"> <li>• Set to true to apply the function to the current subscriber only. If you do not set this parameter, true is the default behavior.</li> <li>• Set to false to apply the function to all subscribers who have the same subscriber ID.</li> </ul>



Table 8: VTA Functions and Input Parameters (*continued*)

sae-set-service-timeout	Sets the service session timeout for the service session identified in the event.	<ul style="list-style-type: none"> <li>• <b>subscription-name</b>—Name of the subscription in the format <i>serviceName%subscriptionId</i>. Default subscriptions have the same name as the service. This parameter is optional when a service-tracking event is being processed. If <i>%subscriptionId</i> is omitted, the default subscription is assumed.</li> <li>• <b>session-name</b>—Name of the service session. This parameter is ignored if a service-tracking event is being processed and the subscription name is omitted. In this case, the session name from the service-tracking event is used. If this parameter is omitted, the default service session is used.</li> <li>• <b>session-timeout</b>—Length of the service session timeout in seconds. If the session timeout is set to 0, the service session is stopped immediately. When the session timeout expires, the service session is stopped.</li> <li>• <b>current-subscriber-only</b>—Specifies whether the function is applied only to the subscriber identified in the event or to all subscribers who have the same subscriber ID. <ul style="list-style-type: none"> <li>• Set to true to apply the function to the current subscriber only. If you do not set this parameter, true is the default behavior.</li> <li>• Set to false to apply the function to all subscribers who have the same subscriber ID.</li> </ul> </li> </ul>
sae-set-user-timeout	Sets the subscriber session timeout of the subscriber identified in the event.	<ul style="list-style-type: none"> <li>• <b>session-timeout</b>—Length of the subscriber session timeout in seconds. If the session timeout is set to 0, the subscriber session is stopped immediately. When the session timeout expires, the subscriber is logged out.</li> <li>• <b>current-subscriber-only</b>—Specifies whether the function is applied only to the subscriber identified in the event or to all subscribers who have the same subscriber ID. <ul style="list-style-type: none"> <li>• Set to true to apply the function to the current subscriber only. If you do not set this parameter, true is the default behavior.</li> <li>• Set to false to apply the function to all subscribers who have the same subscriber ID.</li> </ul> </li> </ul>

Table 8: VTA Functions and Input Parameters (*continued*)

sae-start-service	Starts the specified service subscription.	<ul style="list-style-type: none"> <li>• <b>subscription-name</b>—Name of the subscription in the format <i>serviceName%subscriptionId</i>. Default subscriptions have the same name as the service. This parameter is optional when a service-tracking event is being processed. If <i>%subscriptionId</i> is omitted, the default subscription is assumed.</li> <li>• <b>session-timeout</b>—Length of the service session timeout in seconds. When the session timeout expires, the service session is stopped.</li> <li>• <b>session-name</b>—Name of the service session. If this parameter is omitted, the default service session is used.</li> <li>• <b>current-subscriber-only</b>—Specifies whether the function is applied only to the subscriber identified in the event or to all subscribers who have the same subscriber ID. <ul style="list-style-type: none"> <li>• Set to true to apply the function to the current subscriber only. If you do not set this parameter, true is the default behavior.</li> <li>• Set to false to apply the function to all subscribers who have the same subscriber ID.</li> </ul> </li> <li>• <b>substitution</b>—Policy parameter substitution to use when starting the service. This is specified as <b>substitution name value value</b>. If this parameter is omitted, the service is started without substitutions.</li> <li>• <b>persistent</b>—Specifies whether a service session is persistent. If you use the SRC VTA to activate a service with the persistent option, this service is subsequently activated by the SAE every time the subscriber connects to the network. This option provides efficiency because the SRC VTA does not need to make a decision to activate the service on subsequent logins and because some applications can more efficiently activate a group of services at login. <ul style="list-style-type: none"> <li>• Set to true to cause the session to be persistent.</li> <li>• Set to false to specify that the session is not persistent and the service is activated or deactivated only for the current subscriber session.</li> </ul> </li> </ul>
-------------------	--	--

Table 8: VTA Functions and Input Parameters (*continued*)

sae-stop-service	Stops the specified subscription to the specified service.	<ul style="list-style-type: none"> <li>• <b>subscription-name</b>—Name of the subscription in the format <i>serviceName%subscriptionId</i>. Default subscriptions have the same name as the service. This parameter is optional when a service-tracking event is being processed. If <i>%subscriptionId</i> is omitted, the default subscription is assumed.</li> <li>• <b>session-name</b>—Name of the service session. If the <i>subscriptionName</i> parameter is omitted, this parameter is ignored. If this parameter is omitted, the default service session is used.</li> <li>• <b>reason</b>—Reason for the termination. When the service is stopped, the termination cause can be sent to the billing system so it can differentiate between service stops. If this parameter is omitted, no termination cause is provided to the billing system. Specify an integer that identifies the termination cause. Possible values are defined in RFC 2866—RADIUS Accounting (June 2000).</li> <li>• <b>current-subscriber-only</b>—Specifies whether the function is applied only to the subscriber identified in the event or to all subscribers who have the same subscriber ID. <ul style="list-style-type: none"> <li>• Set to true to apply the function to the current subscriber only. If you do not set this parameter, true is the default behavior.</li> <li>• Set to false to apply the function to all subscribers who have the same subscriber ID.</li> </ul> </li> <li>• <b>persistent</b>—Specifies whether a service session is persistent. <ul style="list-style-type: none"> <li>• Set to true to cause the session to be persistent.</li> <li>• Set to false to specify that the session is not persistent and the service is deactivated only for the subscriber identified in the event.</li> </ul> </li> </ul>
scripts-run-external-script	Executes an external script that is present on the local SRC file system, as previously defined with the <b>shared vta group name processor scripts external-script name</b> configuration statement.	<ul style="list-style-type: none"> <li>• <b>script-name</b>—Name of an external script previously defined with the <b>shared vta group name processor scripts external-script name</b> configuration statement.</li> <li>• <b>name</b>—Any other parameters the script expects to receive as previously defined under the <b>shared vta group name processor scripts external-script name parameters</b> configuration statement. These parameters are entered as name/value pairs.</li> </ul>

Table 8: VTA Functions and Input Parameters (*continued*)

scripts-run-javascript	<p>Executes a script written in the JavaScript programming language that you previously defined by using the <b>shared vta group name processor scripts javascript name</b> configuration statement.</p> <ul style="list-style-type: none"> <li><b>script-name</b>—Name of a script written in the JavaScript programming language that you previously defined by using the <b>shared vta group name processor scripts javascript name</b> configuration statement.</li> </ul> <p>Any parameters referenced in the script itself are taken from the event and passed to the function automatically.</p>
------------------------	---

## On Error

For each action, you must specify what the event handler does if an error occurs; you specify this with the **on-error** option. If an error occurs, the event handler can do one of the following:

- Abort-event-processing—Stop processing the current event.
- Go-to-next-action—Continue with the next action, if any, in the same event handler.
- Go-to-next-event-handler—Skip any remaining actions in the current event handler and proceed to the next event handler (if any).

### Related Documentation

- [How the SRC VTA Works on page 12](#)
- [Overview of Configuring Event Handlers on page 17](#)
- [Configuring Actions for a VTA Event Handler \(SRC CLI\) on page 65](#)
- [Configuring Event Handlers \(SRC CLI\) on page 66](#)

## Overview of Configuring the Event Queue

The following topics provide an overview of configuring the event queue:

- [Setting the Size of the Event Queue on page 28](#)
- [Calculating the Size of the Non-Persistent Event Queue on page 29](#)

### Setting the Size of the Event Queue

The VTA has an event queue that holds plug-in events from the SAE until the VTA processes them. There are two types of event queues as follows:

- With a persistent event queue, no events are lost even if the VTA or the J2EE application server fails and is restarted.
- With a nonpersistent (in memory) event queue, events can be lost if the VTA or the J2EE application server fails. However, for performance reasons you may want to configure a nonpersistent event queue. By default, the queues are configured as nonpersistent event queues.

Set the size of the VTA event queue to about the number of events your VTA can process in 60 seconds.

If you configure a nonpersistent event queue, the event queue size is the maximum number of events that can be lost if the application server or the VTA fails. If the SAE sends events faster than the VTA can process them, the event queue fills, and the VTA signals the SAE to stop sending events for 60 seconds. Your VTAs should be deployed on a host or cluster with sufficient throughput to handle events above the average rate generated by the SAEs in normal operation. This way, the event queue is sufficient to buffer peak event generation rates.



**NOTE:** By default, the maximum RAM size (`java-heap-size`) for the Web application server is set to 1000 MB; 500 MB of this is for the Web application server and 500 MB is for one VTA queue. A VTA event queue with a `max-queue-size` of 100,000 (the default) should not require more than 500 MB of RAM. However, if the queue is configured to be larger, or if multiple VTAs are deployed to the same Web application server, you should increase the maximum RAM size (`java-heap-size`) for the Web application server accordingly. To adjust the RAM for the Web application server see, [“Configuring Local Properties for the Web Application Server \(SRC CLI\)” on page 51](#).

---

## Calculating the Size of the Non-Persistent Event Queue

If communication between the SAE and VTA is lost for an extended period, a large backlog of events can build up in the SAE fail queue. This backlog can result in the SAE sending events far in excess of the average rate for an extended period. Note, however, that smaller event queue sizes affect recovery time if the connection between the SAE and the VTA is disrupted. When the VTA event queue fills, the VTA signals the SAE to stop sending events, and the SAE does not send any events for 60 seconds. If the VTA event queue is limited to less than the number of events the VTA can process in 60 seconds, it will empty the queue and be idle until the SAE starts sending it events again.

For example, if you set the VTA event queue size to the number of events the VTA can process in 30 seconds, after an SAE-VTA communication disruption the SAE rapidly sends events that it has stored in its local fail queue. After the VTA has collected the events that take it 30 seconds to process, it signals the SAE to stop sending events, and the SAE stops for 60 seconds. The VTA will complete processing the events in its queue in 30 seconds and then will be idle for 30 further seconds before the SAE starts to send events again. In this example, the VTA could clear the backlog of events in the SAE's fail queue twice as fast if the VTA's event queue was large enough to hold the number of events it can process in 60 seconds. To configure the maximum size of the event queue (`max-size-byte`), you need to take the number of events the VTA can process in your chosen interval and multiply it by the average size of the plug-in attributes sent to the queues. As a rough estimate, you can use 1024 bytes for twenty plug-in attributes sent to the queue.

Contact the Juniper Technical Assistance Center or Juniper Professional Services if you need further advice about sizing the VTA event queue.

## Overview of Managing VTA Accounts and Sessions

---

The SRC VTA allows service providers to manage accounts and sessions by:

- [Identifying Subscribers, SAEs, and Sessions on page 30](#)
- [Managing VTA Accounts and Sessions on page 30](#)
- [Managing Subscriber Sessions and Service Sessions on page 30](#)

### Identifying Subscribers, SAEs, and Sessions

The SRC VTA must be able to identify each subscriber by a unique identifier. The SRC VTA uses the identifier to manage:

- VTA accounts and sessions
- Subscriber and service sessions

You can configure the SRC VTA to use data keys to identify corresponding data values for these management tasks. The data keys depend on the subscriber's identifier and comprise one or more plug-in attributes. Some identifiers are suitable for residential subscribers and some for enterprise subscribers.

### Managing VTA Accounts and Sessions

Depending on the information that identifies subscribers in your SRC configuration, you can configure the SRC VTA to use several types of plug-in attributes as data keys to identify accounts and sessions in the VTA database. If you use a NIC with the VTA, the SRC VTA can also use some of these plug-in attributes to construct a data key that the NIC can use to determine which SAE manages a subscriber. When the NIC identifies an SAE, the SRC VTA can also obtain a key to identify the subscriber session that the SAE is managing for the subscriber.

You configure the data keys that identify accounts and sessions with the **shared vta group name** statement and specifying the data keys by setting the **subscriber-id-solution** option.

### Managing Subscriber Sessions and Service Sessions

When the SRC VTA receives plug-in events, it may need to start or stop a subscriber session or service session. The plug-in events identify the SAE that manages a subscriber; however, the SRC VTA must construct a data key from one or more plug-in attributes to identify the subscriber session or service session. You configure the data keys that identify subscriber and service sessions with the **shared vta group name** statement. Depending on the information that identifies subscribers in your SRC configuration, you must configure the SRC VTA to use the keys listed in [“Keys Used to Specify the SAE Subscriber ID \(SRC CLI\)” on page 61](#). See [“Creating and Configuring a VTA Shared Group Configuration \(SRC CLI\)” on page 59](#).

#### Related Documentation

- [Overview of the SRC VTA on page 9](#)

- [Configuring a Database to Store Account and Session Data \(SRC CLI\) on page 41](#)
- [Configuring the Initial Balance and Status of a Subscriber Account in the External Database \(SRC CLI\) on page 70](#)
- [Example: Limiting Subscriber Access Based on Account Balances on page 91](#)

## Overview of Adjusting the Interim Accounting Interval

---

When you configure services in the SRC VTA, you can optionally define a formula to dynamically adjust the interim accounting interval for each service based on the subscriber's remaining resources and use of the network for that service. Each service in the SRC VTA can use a different formula. You can configure the SRC VTA software to evaluate the formula to obtain the accounting intervals. Depending on the result, the SRC VTA performs the following functions:

- If the result is zero, the SRC VTA disables interim accounting.
- If the result is a negative number, the SRC VTA does not change the interim accounting interval.
- If the result is a positive number, the SRC VTA changes the interim accounting interval to this value.

The variables used to define the interim accounting interval are categorized as:

- **Current service**—Provides session data of the service for the current service-tracking event.
- **Other service**—Provides service session usage information for another subscriber service for the current service-tracking event. For example, if a subscriber has two quota services, QuotaLocal and QuotaInternet, the interim formula for QuotaLocal can provide usage information to QuotaInternet.
- **Account balance**—Provides the balance in the account.

For details on the variables used to define the interim accounting interval formula, see [“Variables Used to Define the Interim Accounting Interval for Services” on page 72](#).

### Related Documentation

- [Configuring VTA Services and Policies on page 42](#)
- [Variables Used to Define the Interim Accounting Interval for Services on page 72](#)
- [Configuring the Interim Account Interval and Usage Metric of a Service in the External Database \(SRC CLI\) on page 72](#)

## Using JavaScript Programs in VTA Configurations

---

You can use JavaScript programs in your VTA configuration for such tasks as calculating a usage metric or an interim accounting interval, specifying an event condition, updating event attributes in processors, and writing scripts to update accounts.

You can reference a specific set of predefined variables in a JavaScript program, such as plug-in attributes, event attributes, or account balances.

When a variable is referenced or updated in the JavaScript program, enclose it in angle brackets (<>) so that the JavaScript program retrieves only the necessary information. Within the JavaScript program, only one instance of the referenced variable must be enclosed in angle brackets.

You define formulas in the JavaScript scripting language (see <http://wp.netscape.com/eng/mozilla/3.0/handbook/javascript/index.html>). The Quota VTA executes the script in the Rhino JavaScript implementation (see <http://www.mozilla.org/rhino>).

**Related  
Documentation**

- [How the SRC VTA Works on page 12](#)
- [Configuring JavaScript Programs on page 80](#)

---

## Overview of the Web Application Server on C Series Controllers

The SRC software on a C Series Controller includes a Web application server that hosts the Dynamic Service Activator 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.

By default, the SRC Web application server listens on port 8080 for HTTP connections on the **eth0** interface (interface to the trusted network) and on the configured ports for HTTP and HTTPS connections on the **eth1** interface (interface to the untrusted network).

You can control access to applications deployed in the Web application server by configuring virtual hosts. A virtual host contains aliases and lists of the clients that are allowed to access the virtual host.

The aliases are DNS names or IP addresses that appear in the host part of the URLs used by clients to access a Web application. When the Web application server receives a request for an application, it searches for the virtual host with the alias that matches the host in the URL. If a virtual host is found, the Web application server verifies that the application is deployed on this virtual host and the client making the request is allowed to access the virtual host. If no virtual host is found, or if access to the application or client is not allowed by the virtual host, the request is rejected and the client receives an error code.

By default, SRC applications use the virtual host **eth0**. You must configure this virtual host and the following aliases:

- The IP address assigned to **eth0**.
- The name for the SRC host configured at the **[edit system host-name]** and **[edit system domain-name]** hierarchy levels.



For this reason, if you want to access the **eth0** virtual host with URLs containing the DNS name of your SRC host, you must configure your SRC hostname in your DNS server.

You configure the built-in applications, such as Dynamic Service Activator, to deploy the application to a specific virtual host. Other applications that you can load for demonstration purposes are automatically deployed on the built-in virtual host **eth0**.

## Clustering

The SRC Web application server supports clustering, which provides reliability through failover and load balancing. The nodes in the cluster automatically discover one another on startup and automatically synchronize their state with the rest of the group. The cluster configuration is part of the shared SRC configuration and is stored in the Juniper Networks database. You can configure several Web application server clusters. However, a single SRC Web application server instance belongs to only one cluster; it cannot belong to more than one cluster.

### Local and Shared Configuration

If you want a Web application server instance to be part of a cluster, you need to specify the cluster name in the local configuration by using the **[edit slot 0 application-server]** configuration statement. This statement points to the shared configuration stored in the Juniper Networks database. The Web application shared configuration is specified using the **[edit shared application-server cluster *cluster-name*]** configuration statement.

Storing the cluster configuration in the Juniper Networks database ensures that all nodes in the cluster share the same configuration, including the unique identifier of each node, and the shared cluster name. All nodes must be specified within the same Juniper Networks database community name.

The configuration of the application server cluster lists the information about each application server node. When the application server is started, the system retrieves the shared application server cluster configuration and generates the appropriate startup script for the application server node. If no cluster is defined, the application server is started in “all” mode, but without the cluster parameters.



**NOTE:** If you change the shared-cluster configuration, you must restart the local Web application server.

By default, the intra-cluster communication is done through multicasting and UDP is used as the channel stack protocol. If multicasting is not an option for deployment, you can use TCP as the channel stack. The shared cluster configuration is valid only if the following conditions are fulfilled:

- The multicast-address is configured and either the channel stack is not set (the system uses UDP by default) or the channel stack is set to UDP.
- The channel stack is set to TCP and the multicast-address is not configured.

- Related Documentation**
- [Configuring the Web Application Server \(SRC CLI\) on page 49](#)
  - [Configuration Statements for the Web Application Server on page 50](#)

## PART 2

# Configuration

- [Prerequisites for Running the SRC VTA on page 37](#)
- [Configuration Tasks for Using NIC with the SRC VTA on page 45](#)
- [Configuration Tasks for the SRC Web Application Server on page 49](#)
- [Configuration Tasks for the SRC VTA on page 59](#)
- [Examples on page 91](#)



## CHAPTER 3

# Prerequisites for Running the SRC VTA

- [Before You Configure the SRC VTA on page 37](#)
- [Configuring the SAE to Send Tracking Events to the SRC VTA on page 38](#)
- [Configuring the External Database on page 40](#)
- [Configuring VTA Services and Policies on page 42](#)
- [Configuring Subscribers and Subscriptions to VTA Services on page 43](#)

### Before You Configure the SRC VTA

---

Because VTAs rely on other components in the SRC network, you must complete several tasks before you configure the SRC VTA.

Before you configure a VTA, you must complete the following tasks:

1. Deploy a working SRC network.  
  
To support the SRC VTA, you must install SAEs to manage the routers or other devices through which subscribers connect to the network.  
  
[See \*Configuring the SAE \(SRC CLI\)\*.](#)
2. Configure the SRC Web application server.  
  
[See “\*Configuring the Web Application Server \(SRC CLI\)\*” on page 49.](#)
3. (Optional) Configure a NIC that identifies the SAE reference for each subscriber type. You need to complete this task only if your configuration requires a NIC—for example, if an event handler action affects more than just the current subscriber described by an event.  
  
[See “\*Configuring a NIC for the VTA \(SRC CLI\)\*” on page 45.](#)
4. (Optional) Create the Java scripts that VTA invokes.
5. On a separate host, install a relational database to store the data that the SRC VTA tracks. You must have the database server running, and then you must create the VTA database within your database server.  
  
[See, “\*Configuring a Database to Store Account and Session Data \(SRC CLI\)\*” on page 41.](#)

6. To allow the SRC Web application server to connect to your external database, copy the JDBC driver .jar file for your database server brand and version to the `/opt/UMC/appsvr/common/lib` directory on every SRC system running a VTA.

See [“Installing the JDBC Driver .jar File” on page 42.](#)

7. Configure the associated services and policies.

See [“Configuring VTA Services and Policies” on page 42.](#)

8. Configure your subscribers and subscriptions.

See [“Configuring Subscribers and Subscriptions to VTA Services” on page 43 .](#)

9. Configure the SAE to send tracking events to the SRC VTA.

See [“Configuring the SAE to Send Tracking Events to the SRC VTA” on page 38.](#)

#### Related Documentation

- [Overview of the SRC VTA on page 9](#)
- [How the SRC VTA Works on page 12](#)
- [SRC VTA Operation on page 16](#)
- [Overview of the Web Application Server on C Series Controllers on page 32](#)
- [Configuration Statements for the Web Application Server on page 50](#)

---

## Configuring the SAE to Send Tracking Events to the SRC VTA

---

The SRC VTA communicates with the SAE through the Enterprise JavaBean (EJB) adapter plug-in. This plug-in is an SAE plug-in and performs the following functions:

- Filters SAE plug-in events for the SRC VTA.
- Adapts internal SAE events to EJB-compatible methods.
- Sends SAE tracking plug-in events to the SRC VTA.

To configure the EJB adapter plug-in:

1. From configuration mode, access the EJB adapter plug-in configuration. In this sample procedure, the EJB adapter plug-in called QuotaVTA is configured in the nw-area SAE group.

```
user@host# edit shared sae group nw-area configuration plug-ins name QuotaVTA  
ejb-adaptor
```

2. Configure the class name of the J2EE application server's JNDI service provider.

```
[edit shared sae group nw-area configuration plug-ins name QuotaVTA ejb-adaptor]  
user@host# set jndi-service-provider jndi-service-provider
```

3. Configure the URL of the J2EE application server that is running JNDI service.

```
[edit shared sae group nw-area configuration plug-ins name QuotaVTA ejb-adaptor]  
user@host# set application-server-url application-server-url
```

4. Configure the JNDI name of the SAEEventListener EJB of the peer VTA. Because multiple VTA groups are supported, you must specify the SAEEventListenerBean for each VTA group. Specify the SAEEventListenerBean in the following format.:

**vta-VTA group name/SAEEventListenerBean**

For example, if you want a particular SAE to send events to a particular VTA instance (group) called “apple”, set the **jndi-sae-event-listener** as follows:

```
[edit shared sae group nw-area configuration plug-ins name QuotaVTA ejb-adaptor]
user@host# set jndi-sae-event-listener vta-apple/SAEEventListenerBean
```

If you want a particular SAE to send some events to one VTA instance (group) called “apple”, and some events (can be exactly the same events) to another VTA instance called “orange”, configure two separate ejb-adaptor plugins—one with the **jndi-sae-event-listener** set to “vta-apple/SAEEventListenerBean”, and the other with the **jndi-sae-event-listener** set to “vta-orange/SAEEventListenerBean.”

5. (Optional) Configure the LDAP filter that determines the subscriber and service events that the EJB adapter plug-in sends to the VTA. If you specify plug-in attributes in this field, you must include the same attributes in the attributes option.

```
[edit shared sae group nw-area configuration plug-ins name QuotaVTA ejb-adaptor]
user@host# set event-admitter event-admitter
```

Table 9 on page 39 lists the values that you can use for LDAP filter strings.

**Table 9: Settings for Filter Strings**

Filter String	Action
()	Matches no objects
(*)	Matches all objects
<p>List of &lt;attribute&gt;= &lt;value&gt; pairs</p> <p>&lt;attribute&gt;—Name of a property or attribute &lt;ldapAttributeName&gt;</p> <p>&lt;value&gt;—One of the following:</p> <ul style="list-style-type: none"> <li>• * (asterisk)</li> <li>• Explicit string</li> <li>• String that contains an *</li> </ul> <p><b>Note:</b> To define a special character (* &amp; , !   \) in a string, precede it with the backslash symbol (\).</p>	<ul style="list-style-type: none"> <li>• If &lt;value&gt; is *, checks for any value.</li> <li>• If &lt;value&gt; is an explicit string, checks whether any value of the property matches the string, regardless of case.</li> <li>• If &lt;value&gt; is a string that contains a *, checks whether any value of the property contains the string, regardless of case.</li> </ul>
(&<filter><filter>...)	True if all filters match
( <filter><filter>...)	True if at least one filter matches
(!<filter>)	True if the filter does not match

The variables in the filter include the names of plug-in attributes and a `PluginEventType` variable. The value of this variable is the name of the type of event, such as `PE_START_SERVICE`. For names of plug-in attributes and plug-in event types, see the SAE CORBA plug-in documentation on the Juniper Networks Web site at <http://www.juniper.net/techpubs/software/management/src/api-index.html> or in the **SDK+AppSupport+Demos+Samples.tar.gz** file on the Juniper Web site at <https://www.juniper.net/support/products/src/index.html>.

6. (Optional) Specifies whether or not the J2EE application server uses load balancing to determine the location that manages requests to the VTA.

```
[edit shared sae group nw-area configuration plug-ins name QuotaVTA ejb-adaptor]
user@host# set use-ejb-cluster
```

7. Configure load-balancing scheme of the J2EE application server that hosts the VTA. See the documentation for the J2EE application server to determine which load-balancing scheme it supports.

```
[edit shared sae group nw-area configuration plug-ins name QuotaVTA ejb-adaptor]
user@host# set ejb-clustering-strategy (EJBOjectClustering | EJBHomeClustering |
JNDIClustering)
```

8. (Optional) Configure the plug-in attributes that the EJB adapter plug-in sends to the VTA listener. If you do not define a list of attributes, the EJB adapter plug-in sends all plug-in attributes to the VTA. Sending unnecessary plug-in attributes can adversely affect the performance of SRC components.

```
[edit shared sae group nw-area configuration plug-ins name QuotaVTA ejb-adaptor]
user@host# set attributes [(host | router-name | interface-name | ...)...]
```

Specify at least the following plug-in attributes: `router-name`, `session-id`, `login-name`, `user-ip-address`, `ssp-host`, `domain`, `service-name`, `event-time`, `session-time`, `in-octets`, `out-octets`, `in-packets`, `out-packets`, `session-timeout`, `downstream-bandwidth`, `upstream-bandwidth`, `service-session-name`, `subscription-name`. You may need to add attributes if you use them for the event admitter.

- Related Documentation**
- [How the SRC VTA Works on page 12](#)
  - [Installing the JDBC Driver .jar File on page 42](#)

---

## Configuring the External Database

This section describes how to configure an external database to store VTA account and session data.

1. [Configuring a Database to Store Account and Session Data \(SRC CLI\) on page 41](#)
2. [Installing the JDBC Driver .jar File on page 42](#)



## Configuring a Database to Store Account and Session Data (SRC CLI)

The SRC VTA requires a relational database to store accounts and session data. You must create this database on a separate, non-SRC machine before you can run the SRC VTA. For information about databases that we have tested for use with the SRC VTA, see the *SRC Release Notes*.

For each VTA instance, you need to create a database that uses the schema for the SRC VTA. To configure a database:

1. From any SRC Release 4.2 system, navigate to the `/opt/UMC/vta/database/` directory and copy the appropriate VTA schema file and save it on your external database machine. We provide the following VTA schema files:

- `/opt/UMC/vta/database/vta-database-oracle.sql`
- `/opt/UMC/vta/database/vta-database-mysql.sql`

These files contain the SQL statements that you must execute to create a database in either a MySQL or Oracle database server. These files are examples that show the required database schema (in the file for MySQL, you must modify the database username and password before you can execute the SQL statements). If you have a different type of database, you can use these files as a reference on how to create a database with the same schema as is described in these two files.

2. Configure access to the database for an administrator by using the SRC VTA to monitor and manage subscribers. Edit the following options in the file as required for your external database:
  - VTA database name
  - Username
  - Password
3. Use your standard mechanism to execute the SQL statements contained in the file. This creates a database inside your database server, with the schema needed by the SRC VTA.



**NOTE:** If the provided file does not contain SQL sufficient to create a database inside your database server, you must create the database manually. Use the provided file as a reference to understand the exact schema required by the SRC VTA

---

## Installing the JDBC Driver .jar File

To allow the SRC Web application server to connect to your external database, you need to copy the JDBC driver .jar file for your database server brand and version to every SRC system running a VTA.

1. Obtain the relevant .jar file that contains the JDBC driver for your particular database server brand and version.
2. Copy the file to the `/opt/UMC/appsvr/common/lib` directory on every SRC system running a VTA. You can use the SRC CLI “file” commands, FTP, or SCP to copy the file.
3. Restart the Web application server on each SRC system.

```
user@host> restart component appsvr
```

### Related Documentation

- [Configuring the Connection Between the SRC VTA and the External Account and Session Database \(SRC CLI\) on page 62](#)
- [Configuring the Web Application Server \(SRC CLI\) on page 49](#)
- [Overview of the Web Application Server on C Series Controllers on page 32](#)
- [Configuring the Web Application Server Shared Cluster Configuration \(SRC CLI\) on page 52](#)

## Configuring VTA Services and Policies

---

Only the SRC VTA should activate and deactivate services that the SRC VTA controls, and you must ensure that these services are not visible on a portal for subscribers to control manually. You can use other services with a VTA if you design the policies and priorities for those services to work together.

For example, if you manage subscribers with a VTA, you can allow subscribers to manually activate a service that overrides the quota service, and consequently prevents charges in the periodic and bought accounts. You would account for use of this service through RADIUS rather than a VTA, and subscribers would incur an extra cost for using the service. In this case, you configure the overriding service with a higher precedence than the quota service.

To configure services for the SRC VTA:

1. Create services for which a VTA monitors and manages usage.
2. Configure policies that specify ingress and egress accounting rules consistent with the usage formula.

For information about configuring accounting rules for a policy, see [Policy Management Overview](#).

### Related Documentation

- [How the SRC VTA Works on page 12](#)

- [Managing Subscriber Sessions and Service Sessions on page 30](#)
- [Configuring Subscribers and Subscriptions to VTA Services on page 43](#)

## **Configuring Subscribers and Subscriptions to VTA Services**

---

To configure subscribers to VTA services, see the *SRC PE Subscribers and Subscriptions Guide*.

1. Create at least one shared subscriber.
2. For all subscribers managed by the SRC VTA, create an individual or a group subscription to services for which a VTA monitors and manages usage.

### **Related Documentation**

- [Overview of the SRC VTA on page 9](#)
- [Locating the SAE That Manages a Subscriber for the SRC VTA on page 45](#)
- [Configuring VTA Services and Policies on page 42](#)
- [Configuring a Database to Store Account and Session Data \(SRC CLI\) on page 41](#)



## CHAPTER 4

# Configuration Tasks for Using NIC with the SRC VTA

- [Locating the SAE That Manages a Subscriber for the SRC VTA on page 45](#)
- [Configuring a NIC for the VTA \(SRC CLI\) on page 45](#)
- [Configuring NIC Proxies for the VTA on page 46](#)

### Locating the SAE That Manages a Subscriber for the SRC VTA

You can use NIC proxies if the SRC VTA software needs to locate the SAE that manages a particular subscriber. For example, if the VTA receives an account update event and determines that it needs to reconfigure the corresponding SAE session, the VTA must find the SAE that is managing the session. The VTA can do this through the NIC.

You can also use the NIC with the SRC VTA to allow the following:

- Immediately activate subscriptions to quota services—The VTA immediately activates a subscriber's quota service when a deposit is made to the subscriber's account. In this case, the NIC maps the subscriber's identifier to the SAE reference. This scenario is for subscribers who connect to the network through routers running JunosE or Junos OS.

If you do not set up a NIC for this purpose or you use an identifier that the NIC cannot map to an SAE reference, subscribers must log out and log in again before the VTA can activate their quota services when deposits are made to their accounts.

#### **Related Documentation**

- [Identifying Subscribers, SAEs, and Sessions on page 30](#)
- [Configuring Subscribers and Subscriptions to VTA Services on page 43](#)
- [Configuring a NIC for the VTA \(SRC CLI\) on page 45](#)
- [Configuring NIC Proxies for the VTA on page 46](#)

### Configuring a NIC for the VTA (SRC CLI)

For demonstrations and installations with few subscribers, you can configure the VTA to use a NIC proxy stub, which explicitly defines a set of data mappings. However, for most standard installation with a significant number of subscribers and multiple SAEs, you need to set up a full NIC configuration. The requirement to set up a NIC depends on

your VTA configuration. For example, configure a NIC if you specify an action that affects more than just the current subscriber described by an event.

To configure a NIC for the VTA:

1. Use the OnePopLogin configuration scenario (see NIC Configuration Scenarios).
2. Plan and configure the NIC hosts. See Configuring the NIC (SRC CLI).
3. Add the NIC SAE agents to each SAE configuration as external plug-ins. Specify these plug-in attributes: router-name, session-id, user-type, login-name, user-ip-address.

For information about configuring SAE plug-ins, see Configuring the SAE for External Plug-Ins (SRC CLI).

4. (Optional) Configure a NIC proxy stub. See Configuring NIC Test Data (SRC CLI) for information about configuring the NIC proxy stub.
5. Configure a NIC proxy for the VTA. See [“Configuring NIC Proxies for the VTA” on page 46](#).

#### Related Documentation

- Before You Configure the NIC
- [Locating the SAE That Manages a Subscriber for the SRC VTA on page 45](#)
- Starting the NIC (SRC CLI)
- Configuration Statements for the NIC

---

## Configuring NIC Proxies for the VTA

---

For information about NIC proxies, see Overview of NIC Proxy Configuration.

You can configure a NIC proxy that passes the subscriber's identifier to a NIC resolver and receives the corresponding SAE reference. This NIC allows the VTA to immediately activate a subscriber's quota service when a deposit is made to the subscriber's account. This feature is available for subscribers who connect to the network through routers running JunosE or Junos OS.

Use the following statement to configure the NIC proxy for the VTA:

```
shared vta nic-proxy-configuration name
```

To configure the NIC proxy for the VTA:

1. From configuration mode, access the statement that specifies the NIC proxy for the VTA.

```
[edit]
```

```
user@host# edit shared vta nic-proxy-configuration name
```

2. To complete the configuration of the NIC proxy, follow the procedure described in chapter *Configuring SRC Applications to Communicate with an SAE (SRC CLI)* in the *SRC PE Network Guide*.

3. Create a reference between the shared VTA group configuration and the NIC proxy configuration.

From the **[edit shared vta group *name*]** hierarchy level execute:

```
[edit shared vta group name]  
user@host# set nic-proxy nic-proxy
```

Set the **nic-proxy** option to name you specified in Step 1. For more information, see [“Creating and Configuring a VTA Shared Group Configuration \(SRC CLI\)” on page 59](#).

**Related  
Documentation**

- [Before You Configure a NIC Proxy](#)
- [Locating the SAE That Manages a Subscriber for the SRC VTA on page 45](#)
- [Configuring a NIC for the VTA \(SRC CLI\) on page 45](#)





## CHAPTER 5

# Configuration Tasks for the SRC Web Application Server

- [Configuring the Web Application Server \(SRC CLI\) on page 49](#)
- [Configuration Statements for the Web Application Server on page 50](#)
- [Configuring Local Properties for the Web Application Server \(SRC CLI\) on page 51](#)
- [Configuring the Web Application Server Shared Cluster Configuration \(SRC CLI\) on page 52](#)
- [Configuring the Nodes in the Web Application Server Cluster \(SRC CLI\) on page 53](#)
- [Configuring Remote Access to the Application Server \(SRC CLI\) on page 54](#)
- [Configuring Virtual Hosts for the Web Applications \(SRC CLI\) on page 55](#)
- [Configuring User Accounts for Web Applications \(SRC CLI\) on page 56](#)
- [Installing Web Applications in the Application Server on page 57](#)
- [Starting the Web Application Server on a C Series Controller on page 58](#)

## Configuring the Web Application Server (SRC CLI)

---

Tasks to configure the Web application server are:

1. [Configure the Web application server shared cluster configuration.](#)  
[See “Configuring the Web Application Server Shared Cluster Configuration \(SRC CLI\)” on page 52.](#)
2. [Configure the operating properties.](#)  
[See “Configuring Local Properties for the Web Application Server \(SRC CLI\)” on page 51.](#)
3. [Configure the nodes of the Web application server cluster.](#)  
[See “Configuring the Nodes in the Web Application Server Cluster \(SRC CLI\)” on page 53.](#)
4. [Configure remote access to the application server.](#)  
[See “Configuring Remote Access to the Application Server \(SRC CLI\)” on page 54.](#)

5. Configure the virtual host for the Web application, including whether to allow or deny access by specific remote clients.

See “[Configuring Virtual Hosts for the Web Applications \(SRC CLI\)](#)” on page 55.

6. Configure the user accounts for the Web application.

See “[Configuring User Accounts for Web Applications \(SRC CLI\)](#)” on page 56.

**Related  
Documentation**

- [Overview of the Web Application Server on C Series Controllers on page 32](#)
- [Configuring the Web Application Server Shared Cluster Configuration \(SRC CLI\) on page 52](#)
- [Configuring the Nodes in the Web Application Server Cluster \(SRC CLI\) on page 53](#)

---

## Configuration Statements for the Web Application Server

---

Use the following configuration statements to configure the operating properties for the Web application server at the **[edit]** hierarchy level.

```
slot number application-server {
    java-garbage-collection-options java-garbage-collection-options;
    java-heap-size java-heap-size;
    shared-cluster shared-cluster
}

shared application-server cluster name {
    channel-stack (udp|tcp);
    multicast-address multicast-address;
}

shared application-server cluster name nodes node address {
    node-id node-id;
}

slot number application-server web http {
    port port;
    interface interface;
}

slot number application-server web https {
    local-certificate local-certificate;
    port port;
    interface interface;
}

slot number application-server web virtual-host host-name {
    alias alias;
    allow-address allow-address;
    allow-host allow-host;
    deny-address deny-address;
    deny-host deny-host;
}

shared application-server user name

shared application-server user name authentication {
```

```

encrypted-password encrypted-password;
plain-text-password;
}

```

#### Related Documentation

- [Configuring the Web Application Server \(SRC CLI\) on page 49](#)
- [Configuring Remote Access to the Application Server \(SRC CLI\) on page 54](#)
- [Configuring Virtual Hosts for the Web Applications \(SRC CLI\) on page 55](#)
- [Configuring User Accounts for Web Applications \(SRC CLI\) on page 56](#)
- [Overview of the Web Application Server on C Series Controllers on page 32](#)

## Configuring Local Properties for the Web Application Server (SRC CLI)

To configure basic local properties:

1. From configuration mode, access the configuration statement that configures the local properties.

```
user@host# edit slot 0 application-server
```

2. Configure the garbage collection functionality of the Java Virtual Machine.

```
[edit slot 0 application-server]
```

```
user@host# set java-garbage-collection-options java-garbage-collection-options
```

3. (Optional) If you encounter problems caused by lack of memory, change the maximum memory size available to the JRE.

```
[edit slot 0 application-server]
```

```
user@host# set java-heap-size java-heap-size
```

4. (Optional) Configure the cluster name. Specify the shared-cluster as */application-server/shared-cluster*.

```
[edit slot 0 application-server]
```

```
user@host# set shared-cluster /application-server/shared-cluster
```

For example, to configure a shared cluster called cluster-1:

```
[edit slot 0 application-server]
```

```
user@host# set shared-cluster /application-server/cluster-1
```



**NOTE:** If you change the shared cluster name, you must restart the local application server for the change to take effect.

5. (Optional) Verify your configuration.

```
[edit slot 0 application-server]
```

```
user@host# show
```

```

shared-cluster /application-server/cluster-1;
web {
  http {
    interface eth0;
    port 8080;
  }
}

```

```
    }  
    virtual-host eth0;  
}
```

**Related  
Documentation**

- [Configuring the Web Application Server Shared Cluster Configuration \(SRC CLI\) on page 52](#)
- [Configuring the Nodes in the Web Application Server Cluster \(SRC CLI\) on page 53](#)
- [Overview of the Web Application Server on C Series Controllers on page 32](#)

---

## Configuring the Web Application Server Shared Cluster Configuration (SRC CLI)

Use the following statements to configure a Web application server shared cluster configuration:

```
shared application-server cluster name {  
    channel-stack (udp|tcp);  
    multicast-address multicast-address;  
}
```

To configure the Web application server shared cluster configuration:

1. From configuration mode, access the statement that configures the shared cluster configuration. The name you specify must match the name you configured for the local configuration at the **[edit slot 0 application-server]** hierarchy level.

```
user@host# edit shared application-server cluster name
```

For example, if you have the following local configuration:

```
[edit slot 0 application-server]  
shared-cluster /application-server/cluster-1
```

You need to specify cluster-1 as the cluster name for the shared configuration:

```
user@host# edit shared application-server cluster cluster-1
```

2. Configure the channel stack.

```
[edit shared application-server cluster cluster-1]  
user@host# set channel-stack (udp|tcp)
```

3. (Optional) Specify the multicast address. The multicast address is required only if UDP is selected as the channel stack.

```
[edit shared application-server cluster cluster-1]  
user@host# set multicast-address multicast-address
```

4. (Optional) Verify your configuration.

```
[edit shared application-server cluster cluster-1]  
user@host# show  
  
channel-stack tcp;  
[edit shared application-server cluster cluster-1]  
user@host#
```

- Related Documentation**
- [Overview of the Web Application Server on C Series Controllers on page 32](#)
  - [Configuring the Nodes in the Web Application Server Cluster \(SRC CLI\) on page 53](#)
  - [Configuring the Web Application Server \(SRC CLI\) on page 49](#)
  - [Configuring Local Properties for the Web Application Server \(SRC CLI\) on page 51](#)
  - [Viewing the Web Application Server Cluster Status \(SRC CLI\)](#)

## Configuring the Nodes in the Web Application Server Cluster (SRC CLI)

Use the following statements to configure the nodes in the Web application server cluster:

```
shared application-server cluster name nodes node address {
  node-id node-id;
}
```

To configure the Web application server cluster nodes:

1. From configuration mode, access the statement that configures the cluster nodes and specify the IP address of the node.

```
user@host# shared application-server cluster name nodes node address {
```

2. Configure the node ID for the node. The node ID is a random number you assign to the node. Each node must have a unique node ID specified as the integer type.

```
[edit shared application-server cluster name nodes node address]
user@host# set node-id node-id
```

3. (Optional) Verify your configuration.

```
[edit shared application-server cluster name nodes node address]
user@host# show
```

Following is a sample output of the cluster node configuration:

```
channel-stack udp;
multicast-address 255.255.100.100;
nodes {
  node 10.1.2.3 {
    node-id 2;
  }
  node 10.1.2.4 {
    node-id 1;
  }
  node 10.1.2.5 {
    node-id 4;
  }
}
```

- Related Documentation**
- [Overview of the Web Application Server on C Series Controllers on page 32](#)
  - [Configuring the Web Application Server Shared Cluster Configuration \(SRC CLI\) on page 52](#)
  - [Configuring the Web Application Server \(SRC CLI\) on page 49](#)

- [Configuring Local Properties for the Web Application Server \(SRC CLI\) on page 51](#)

## Configuring Remote Access to the Application Server (SRC CLI)

---

Before you can start using the application server, you need to configure and enable access to the application server. You can make the application server accessible through secure HTTP (HTTPS) or HTTP.

- [Configuring Access to the Application Server Through Secure HTTP on page 54](#)
- [Configuring Access to the Application Server Through HTTP on page 54](#)

### Configuring Access to the Application Server Through Secure HTTP

Before you configure access to the application server through HTTPS, obtain a digital security certificate on the system.

To make the application server accessible through HTTPS:

1. From configuration mode, access the statement that configures access through HTTPS.

```
user@host# edit slot 0 application-server web https
```

2. Specify which TCP port is to receive incoming connection requests for the application server.

```
[edit slot 0 application-server web https]  
user@host# set port port
```

3. Specify the interface to be used for connections to the application server.

```
[edit slot 0 application-server web https]  
user@host# set interface interface
```

On a C Series Controller, use **eth1** for built-in Web applications; you can use **eth0** for demonstration applications.

4. Specify the name of the certificate on the local system.

```
[edit slot 0 application-server web https]  
user@host# set local-certificate local-certificate
```

5. (Optional) Configure user accounts to allow specified clients to authenticate with the application server.

### Configuring Access to the Application Server Through HTTP

To make the application server accessible through HTTP:

1. From configuration mode, access the statement that configures access through HTTP.

```
user@host# edit slot 0 application-server web http
```

2. Specify which TCP port is to receive incoming connection requests for the application server.

```
[edit slot 0 application-server web http]
```

```
user@host# set port port
```

3. Specify the interface to be used for connections to the application server.

```
[edit slot 0 application-server web http]
```

```
user@host# set interface interface
```

On a C Series Controller, use **eth1** for built-in Web applications; you can use **eth0** for demonstration applications.

4. (Optional) Configure user accounts to allow specified clients to authenticate with the application server.

#### Related Documentation

- [Configuring the Web Application Server \(SRC CLI\) on page 49](#)
- [Configuring User Accounts for Web Applications \(SRC CLI\) on page 56](#)
- [Overview of the Web Application Server on C Series Controllers on page 32](#)
- [Overview of Digital Certificates](#)

## Configuring Virtual Hosts for the Web Applications (SRC CLI)

Use the following configuration statements to configure virtual hosts at the **[edit]** hierarchy level:

```
slot number application-server web virtual-host host-name {
  alias [alias...];
  allow-address [allow-address...];
  allow-host [allow-host...];
  deny-address [deny-address...];
  deny-host [deny-host...];
}
```

To configure virtual hosts for the Web applications:

1. From configuration mode, access the statement that configures the virtual host.

By default, SRC applications use the virtual host **eth0**. You must configure this virtual host.

```
user@host# edit slot 0 application-server virtual-host eth0
```

By default, SRC applications run on virtual host **eth0**. You must configure **eth0** as a virtual host. The hostname must be unique.

2. Specify the alternate DNS names or IP addresses for the virtual host.

```
[edit slot 0 application-server virtual-host eth0]
```

```
user@host# set alias [alias ...]
```

The alias must be unique. Specify the following alias for the **eth0** virtual host:

- The IP address assigned to **eth0**.
- The name for the SRC host configured at the **[edit system host-name]** and **[edit system domain-name]** hierarchy levels.

3. Configure access to the virtual host. Specify the IP addresses for remote clients that are allowed access to the virtual host.

```
[edit slot 0 application-server virtual-host eth0]
user@host# set allow-address [allow-address...]
```

4. Configure access to the virtual host. Specify the hostnames for remote clients that are allowed access to the virtual host.

```
[edit slot 0 application-server virtual-host eth0]
user@host# set allow-host [allow-host...]
```

5. Deny access to the virtual host. Specify the IP addresses for remote clients that are denied access to the virtual host.

```
[edit slot 0 application-server virtual-host eth0]
user@host# set deny-address [deny-address...]
```

6. Deny access to the virtual host. Specify the hostnames for remote clients that are denied access to the virtual host.

```
[edit slot 0 application-server virtual-host eth0]
user@host# set deny-host [deny-host...]
```

#### Related Documentation

- [Configuring the Web Application Server \(SRC CLI\) on page 49](#)
- [Overview of the Web Application Server on C Series Controllers on page 32](#)

## Configuring User Accounts for Web Applications (SRC CLI)

User accounts provide one way for clients to authenticate with the application server. For each account, you define the login name for the user, authentication information, and role. You can configure plain-text password or encrypted password as the type of authentication for user accounts. When you delete user accounts, the software verifies that the user account is not referenced by another configuration.



**NOTE:** Client profiles can be cached by applications for 30 minutes. If you change the password or role of a client that has been used within the last 30 minutes, it can take up to 30 minutes before these changes take effect.

If you do not want to wait 30 minutes for the changes to take effect, restart the Web application server.

Use the following configuration statements to configure user accounts at the **[edit]** hierarchy level:

```
shared application-server user name
shared application-server user name authentication {
  encrypted-password encrypted-password;
  plain-text-password;
  role [DSA | PCMM | VTA-group name;
}
```



To configure a user account:

1. From configuration mode, access the configuration statement that configures a user account and specify a username that identifies the client.

```
user@host# edit shared application-server user name
```

The username must be unique within the system. Do not include spaces, colons, or commas in the username.

2. Configure authentication for the user account.

```
[edit shared application-server user name]
user@host# set authentication (plain-text-password | encrypted-password)
```

where:

- **plain-text-password**—Prompt for a plain-text (unencrypted) password.
- **encrypted-password**—Password encoded with crypt. The format of encrypted passwords is "{crypt}<13-characters in a-zA-Z0-9./>".

We recommend that you not enter the password in encrypted format.

For example:

```
user@host# set authentication plain-text-password
New password: type password here
Retype new password: retype password here
```

3. Configure the role for the user account.

```
[edit shared application-server user name]
user@host# set role VTA-Quota
```

Set the role to one of the following values:

- **DSA**—Role for clients accessing the DSA services `dsa-service` and `dsa2-service`
- **PCMM**—Role for clients accessing the DSA service `pcmm-service`
- **VTA-group name**—Role for clients accessing the SOAP API for VTA. The CLI returns all VTA groups configured under the **[edit shared vta group]** edit hierarchy with the prefix "VTA". For example, set the role to VTA-Quota for clients accessing the SOAP API for the VTA group called Quota.

#### Related Documentation

- [Configuring Remote Access to the Application Server \(SRC CLI\) on page 54](#)
- [API for Volume Tracking Application](#)
- [Enabling the SOAP Interface for a VTA Group \(SRC CLI\)](#)
- [Methods for the Volume Tracking Application SOAP Interface](#)

## Installing Web Applications in the Application Server

The SRC software includes a Web application server component for deploying Web applications. You can also deploy a Web application inside JBoss.

We recommend you deploy Web applications under the SRC Web application server by following this procedure.

If you need to deploy a Web application inside JBoss, see the [\[Unresolved xref\]](#) procedure in the *SRC PE Sample Applications Guide*.



**NOTE:** You can deploy a Web application in the Web application server for lab tests and demonstrations. However, running non-SRC Web applications in production environments is not supported.

To deploy a Web application in the SRC Web application server:

1. Start the Web application server.
2. Prepare the Web application archive (WAR) file on a machine other than the C Series Controller.
3. Deploy the WAR file on the C Series Controller. The application server automatically starts the Web application when a new WAR file is deployed.

```
user@host> request appsvr deploy file name
```

For example:

```
user@host> request appsvr deploy file ftp://host/path/ssportal.war
```

**Related  
Documentation**

- Removing Web Applications from the Application Server
- [Starting the Web Application Server on a C Series Controller on page 58](#)
- Restarting the Web Application Server on a C Series Controller

---

## Starting the Web Application Server on a C Series Controller

To start the Web application server on a C Series Controller:

```
user@host> enable component appsvr
```

**Related  
Documentation**

- Restarting the Web Application Server on a C Series Controller
- Stopping the Web Application Server on a C Series Controller

## CHAPTER 6

# Configuration Tasks for the SRC VTA

- [Configuring a VTA Shared Group Configuration \(SRC CLI\) on page 59](#)
- [Configuring the Connection Between the SRC VTA and the External Account and Session Database \(SRC CLI\) on page 62](#)
- [Configuring Actions for a VTA Event Handler \(SRC CLI\) on page 65](#)
- [Configuring Event Handlers \(SRC CLI\) on page 66](#)
- [Configuring the Event Queue \(SRC CLI\) on page 68](#)
- [Configuring the DB-Engine Processor for the SRC VTA Group \(SRC-CLI\) on page 69](#)
- [Configuring the Mailer Processor for the SRC VTA Group \(SRC CLI\) on page 77](#)
- [Configuring the SRC VTA Scripts Processor \(SRC CLI\) on page 79](#)
- [Configuring VTA Logging \(SRC CLI\) on page 83](#)
- [Enabling, Disabling, and Restarting the VTA \(SRC CLI\) on page 88](#)
- [Using One VTA Account for Multiple Subscriber Sessions on page 89](#)

## Configuring a VTA Shared Group Configuration (SRC CLI)

---

- [Creating and Configuring a VTA Shared Group Configuration \(SRC CLI\) on page 59](#)
- [Keys Used to Specify the Subscriber ID Solution \(SRC CLI\) on page 60](#)
- [Keys Used to Specify the SAE Subscriber ID \(SRC CLI\) on page 61](#)

## Creating and Configuring a VTA Shared Group Configuration (SRC CLI)

You can set up multiple VTAs; each VTA is configured as a separate VTA group with its own shared configuration.

Use the following statements to configure a VTA shared configuration:

```
shared vta group name {  
    subscriber-id-solution subscriber-id-solution ;  
    nic-proxy nic-proxy;  
}
```

To configure a VTA shared configuration:

1. From configuration mode, access the statement that configures a VTA shared group configuration. For example, to configure a VTA group called `vta1`:

```
[edit]
user@host# edit shared vta group vta1
```

2. Specify the **subscriber-id-solution** option. This option specifies a data key that uniquely identifies subscriber accounts and sessions in the external database. Some settings also provide information that the NIC and the SAE use to identify subscribers.

```
[edit shared vta group vta1]
user@host# set subscriber-id-solution subscriber-id-solution
```

For more information about configuring the **subscriber-id-solution** option, see [“Keys Used to Specify the Subscriber ID Solution \(SRC CLI\)” on page 60](#).

3. Specify the location of the NIC proxy configuration relative to the configuration properties for the SRC VTA.

```
[edit shared vta group vta1]
user@host# set nic-proxy nic-proxy
```

Set this option to the name you configured under the **[edit shared vta nic-proxy-configuration *name*]** configuration statement. See [“Configuring NIC Proxies for the VTA” on page 46](#).

If you are using a NIC to map subscriber identifiers to an SAE, and you select a VTA subscriber ID value that provides a data key for the NIC, specify the NIC proxy that uses that data key.

4. Verify your configuration.

```
[edit shared vta group vta1]
user@host# show
```

5. Commit your configuration.

```
[edit shared vta group vta1]
user@host# commit
```



**NOTE:** You must enable the new VTA instance with the enable component **vta-name** command.

---

## Keys Used to Specify the Subscriber ID Solution (SRC CLI)

You use the **subscriber-id-solution** option under the **shared vta group *name*** configuration statement to specify keys for managing accounts and sessions. [Table 10 on page 61](#) shows the keys that you can specify for the SRC VTA to query the VTA database, NIC, and SAE. For the SRC VTA to use a subscriber identifier, the plug-in event must include the corresponding NIC or SAE attributes that are listed in [Table 10 on page 61](#) (attributes that start with PA\_). For more information about plug-in attributes, see the documentation for the SAE CORBA plug-in on the Juniper Networks Web site at

<http://www.juniper.net/techpubs/software/management/src/api-index.html>

**Table 10: Keys That the SRC VTA Constructs to Manage Accounts and Sessions**

Subscriber's Identifier	Database Key	Corresponding NIC Key	Corresponding SAE Key
<b>accounting-id</b>	PA_ACCOUNTING_ID	PA_ACCOUNTING_ID	IP address that is returned from the NIC lookup
<b>user-dn</b>	PA_USER_DN	PA_USER_DN	PA_USER_DN
<b>interface-alias</b>	PA_INTERFACE_ALIAS	None	None
<b>interface-alias-and-router</b>	PA_INTERFACE_ALIAS@PA_ROUTER_NAME	PA_ROUTER_NAME	None
<b>interface-and-router</b>	PA_INTERFACE_NAME@PA_ROUTER_NAME	PA_ROUTER_NAME	PA_INTERFACE_NAME
<b>login-name</b>	PA_LOGIN_NAME	PA_LOGIN_NAME	PA_LOGIN_NAME (default)
<b>mac-address</b>	PA_USER_MAC_ADDRESS	None	None
<b>primary-user-name</b> (PPP login name or public DHCP name)	PA_PRIMARY_USER_NAME	None	PA_PRIMARY_USER_NAME
<b>nas-port-id-and-router</b>	PA_PORT_ID@PA_ROUTER_NAME	None	None

**login-name**, **user-dn**, and **interface-and-router** also provide data keys for the NIC and the SAE; the other settings do not.

### Keys Used to Specify the SAE Subscriber ID (SRC CLI)

Depending on the information that identifies subscribers in your SRC configuration, you must configure the SRC VTA to use the keys shown in [Table 11 on page 61](#).

**Table 11: Keys That the SRC VTA Constructs to Manage Subscriber and Service Sessions**

Subscriber's Identifier	SAE Key
<b>user-dn</b>	PA_USER_DN
<b>interface-and-router</b>	A combination of the following plug-in attributes: <ul style="list-style-type: none"> <li>PA_INTERFACE_NAME</li> <li>PA_ROUTER_NAME</li> </ul>

Table 11: Keys That the SRC VTA Constructs to Manage Subscriber and Service Sessions *(continued)*

Subscriber's Identifier	SAE Key
<b>user-ip</b>	A combination of the following plug-in attributes: <ul style="list-style-type: none"> <li>• PA_USER_IP_ADDRESS</li> <li>• PA_EVENT_TIME</li> </ul>
<b>user-ip-and-interface-and-router</b> (IP address of the subscriber on an interface)	A combination of the following plug-in attributes: <ul style="list-style-type: none"> <li>• PA_USER_IP_ADDRESS</li> <li>• PA_INTERFACE_NAME</li> <li>• PA_ROUTER_NAME</li> </ul>
<b>login-name</b>	PA_LOGIN_NAME
<b>primary-user-name</b> (PPP login name or public DHCP name)	PA_PRIMARY_USER_NAME
<b>user-session-id</b>	PA_USER_SESSION_ID

#### Related Documentation

- [Overview of Managing VTA Accounts and Sessions on page 30](#)
- [Overview of the SRC VTA on page 9](#)
- [Using One VTA Account for Multiple Subscriber Sessions on page 89](#)
- [Managing VTA Accounts and Sessions on page 30](#)

## Configuring the Connection Between the SRC VTA and the External Account and Session Database (SRC CLI)

The SRC VTA requires a relational database to store accounts and session data. The database connection information specifies how the VTA connects to this database. You must configure the connection to the external database for each VTA group.



**NOTE:** The values shown in the following sample procedure should work for a MySQL database. The values are different for an Oracle database, or other database servers. The values may also differ for different versions of the same brand of database server. You must understand how to connect to your particular brand and version of database server using JDBC before configuring the database connection.

Following is an example Oracle configuration:

```
user@host> show configuration shared vta group test database
check-valid-connection-sql 'select 1 from dual';
connection-url jdbc:oracle:thin:@//10.10.1.2:1521/vta;
datasource-mapping Oracle9i;
driver-class oracle.jdbc.OracleDriver;
max-pool-size 50;
min-pool-size 5;
password *****;
user-name vta;

user@host>
```

Use the following statements to configure the connection to the external database:

```
shared vta group name database {
  connection-url connection-url;
  driver-class driver-class;
  user-name user-name;
  password password;
  data-source-mapping data-source-mapping;
  min-pool-size min-pool-size;
  max-pool-size max-pool-size;
  check-valid-connection-sql check-valid-connection-sql;
}
```

To configure the connection to the external database:

1. From configuration mode, access the statement that configures the connection to the external database. For example, to configure the connection for a VTA group called vta1:

```
[edit]
user@host# edit shared vta group vta1 database
```

2. Configure the URL connection information. For example, to specify the connection url as jdbc:mysql://10.10.10.1:3306/vta1:

```
[edit shared vta group vta1 database]
user@host# set connection-url jdbc:mysql://10.10.10.1:3306/vta1
```

3. Configure the driver class information. For example, to configure a JDBC driver for MySQL:

```
[edit shared vta group vta1 database]
user@host# set driver-class com.mysql.jdbc.Driver
```

4. Configure the username used to access the database. For example to configure the username called `admin1`:

```
[edit shared vta group vta1 database]
user@host# set user-name admin1
```

5. Configure the password used to access the database. For example, to configure the password as `pwd1`:

```
[edit shared vta group vta1 database]
user@host# set password pwd1
```

6. Configure the data source mapping information. For example, for MySQL:

```
[edit shared vta group vta1 database]
user@host# set data-source-mapping mysql
```

7. Specify the minimum number of simultaneous connections to the database.

```
[edit shared vta group vta1 database]
user@host# set min-pool-size min-pool-size ;
```

8. Specify the maximum number of simultaneous connections to the database.

```
[edit shared vta group vta1 database]
user@host# set max-pool-size max-pool-size ;
```



**NOTE:** The `max-concurrency` option should be set greater than or equal to the `max-pool-size` option; otherwise, database connections are likely to be unused, see [“Overview of Configuring Event Handlers” on page 17](#).

---

9. Specify the SQL used to check whether a database connection is still valid.

```
[edit shared vta group vta1 database]
user@host# check-valid-connection-sql check-valid-connection-sql ;
```

The specified SQL depends on the type of database server. For example, with Oracle, you can use “select 1 from dual,” and with MySQL or MS SQL Server, you can use “select 1”.

#### Related Documentation

- [Configuring the External Database on page 40](#)
- [Configuring the Web Application Server \(SRC CLI\) on page 49](#)
- [Managing VTA Accounts and Sessions on page 30](#)
- [Troubleshooting Database Deadlocks on page 109](#)



## Configuring Actions for a VTA Event Handler (SRC CLI)

Use the following statement to configure an action that can be invoked by one or more event handlers:

```
shared vta group name action action-name {
  function (db-engine-calculate-interim | db-engine-calculate-usage |
    db-engine-get-accounts | db-engine-terminate-session | db-engine-update-accounts
    | mailer-send | sae-set-interim-interval | sae-set-service-timeout | sae-set-user-timeout
    | sae-start-service | sae-stop-service | scripts-run-external-script |
    scripts-run-javascript);
  on-error (abort-event-processing | go-to-next-action | go-to-next-event-handler);
  parameter;
}
```



**NOTE:** We recommend that when you configure event handlers and their actions, you ensure that for any given event, all database operations are performed before any other operations that have permanent effects. This is because if a database error occurs—for example, due to normal contention for database records between different event threads—the VTA rolls back the current database transaction (no changes are made to the database) and then restarts processing the event. If the event performs some other operation other than database operations before such an error, such as start a service, then that other operation is performed again when the event is reprocessed following the error.

1. From configuration mode, access the statement that configures an action and specify a name for the action. For example, to configure an action called `act1` in the VTA group called `vta1`:

```
[edit]
user@host# edit shared vta group vta1 action atc1
```

2. Specify a function you want the action to call. For example, to have the action call the **db-engine-calculate-usage** function:

```
[edit shared vta group vta1 action atc1]
user@host# set function db-engine-calculate-usage
```

3. (Optional) Specify any input parameters required for the function.

```
[edit shared vta group vta1 action atc1]
user@host# set parameter
```

The parameters that you need to set depend on what function you specified. Some functions require no input parameters. You can use the “?” completer to see what parameters need to be set, or see [“Overview of Configuring Actions” on page 21](#) for a complete list of functions and their associated input parameters.

4. Specify what you want the action to do in response to an error. For example, to configure the action to abort processing if an error occurs:

```
[edit shared vta group vta1 action atc1]
user@host# set on-error abort-event-processing
```

5. Verify your configuration.

```
[edit shared vta group vta1 action atc1]
user@host# show

user@host# show
function db-engine-calculate-usage;
on-error abort-event-processing;

[edit shared vta group vta1 action atc1]
user@host#
```

6. Commit your configuration.

```
[edit shared vta group vta1 action atc1]
user@host# commit
```

#### Related Documentation

- [Overview of Configuring Actions on page 21](#)
- [Overview of Configuring Event Handlers on page 17](#)
- [Configuring Event Handlers \(SRC CLI\) on page 66](#)

---

## Configuring Event Handlers (SRC CLI)

Use the following statements to configure event handlers:

```
shared vta group name event-handler event-handler-name {
  events events;
  actions actions;
  condition condition;
  priority priority;
}
```

To configure an event handler:



.....

**NOTE:** We recommend that when you configure event handlers and their actions, you ensure that for any given event, all database operations are performed before any other operations that have permanent effects. This is because if a database error occurs—for example, due to normal contention for database records between different event threads—the VTA rolls back the current database transaction (no changes are made to the database) and then restarts processing the event. If the event performs some other operation other than database operations before such an error, such as start a service, then that other operation is performed again when the event is reprocessed following the error.

.....

1. From configuration mode, access the statement that configures a VTA event handler. For example, to configure an event handler called evh1 in the VTA group called vta1:

```
[edit]
user@host# edit shared vta group vta1 event-handler evh1
```

2. Specify an event. For example, to have the event handler handle start events for the service named QuotaInternet:

```
[edit shared vta group vta1 event-handler evh1]
user@host# set events service-start:QuotaInternet
```

3. Specify previously configured actions that the event handler invokes in response to the event. For example, to specify the previously configured actions called act1, act2, and act3:

```
[edit shared vta group vta1 event-handler evh1]
user@host# set actions [act1 act2 act3]
```

4. Specify the condition the event handler uses to determine whether it should handle the event.

```
[edit shared vta group vta1 event-handler evh1]
user@host# set condition condition
```

Specify the condition as script written in the JavaScript programming language that must return one of the following Boolean values:

- True—Event handler should handle the event.
- False—Event handler should not handle the event.

5. Specify the priority of the event handler.

```
[edit shared vta group vta1 event-handler evh1]
user@host# set priority priority
```

Event handlers with the lower number priority have the highest priority and evaluate the event first.

6. Verify your configuration.

```
[edit shared vta group vta1 event-handler evh1]
user@host# show
```

#### Related Documentation

- [How the SRC VTA Works on page 12](#)
- [Overview of Configuring Actions on page 21](#)
- [Overview of Configuring Event Handlers on page 17](#)
- [Configuring Actions for a VTA Event Handler \(SRC CLI\) on page 65](#)

## Configuring the Event Queue (SRC CLI)

Use the following statement to configure the event queue:

```
shared vta group name queue {
  max-concurrency max-concurrency;
  max-queue-size max-queue-size;
  persistent;
}
```

To configure the event queue:

1. From configuration mode, access the statement that configures the event queue. For example, to configure the event queue for the VTA group called `vta1`:

```
[edit]
user@host# edit shared vta group vta1 queue
```

2. Specify the maximum number of threads consuming events from the queue and passing them to the configured event handlers.

```
[edit shared vta group vta1 queue]
user@host# set max-concurrency max-concurrency
```



**NOTE:** `max-concurrency` should be set greater than or equal to the `max-pool-size` option; otherwise, database connections are likely to be unused, see [“Configuring the Connection Between the SRC VTA and the External Account and Session Database \(SRC CLI\)”](#) on page 62.

3. Specify the maximum queue size.

```
[edit shared vta group vta1 queue]
user@host# set max-queue-size max-queue-size
```



**NOTE:** By default, the maximum RAM size (`java-heap-size`) for the Web application server is set to 1000 MB; 500 MB of this is for the Web application server and 500 MB is for one VTA queue. A VTA event queue with a `max-queue-size` of 100,000 (the default) should not require more than 500 MB of RAM. However, if the queue is configured to be larger, or if multiple VTAs are deployed to the same Web application server, you should increase the maximum RAM size (`java-heap-size`) for the Web application server accordingly. To adjust the RAM for the Web application server see, [“Configuring Local Properties for the Web Application Server \(SRC CLI\)”](#) on page 51.

4. (Optional) Specify whether the event queue is persistent (events are saved to disk) or non-persistent (events are saved in memory). If set, events are saved to disk. By default, the event queue is set to non-persistent.

```
[edit shared vta group vta1 queue]
user@host# set persistent
```

- Related Documentation**
- [Setting the Size of the Event Queue on page 28](#)
  - [Calculating the Size of the Non-Persistent Event Queue on page 29](#)
  - [Overview of Configuring Event Handlers on page 17](#)

## Configuring the DB-Engine Processor for the SRC VTA Group (SRC-CLI)

The SRC VTA uses the db-engine processor to update database and subscriber accounts. The db-engine processor works as a proxy to the external database. It calculates usage, updates account balances, retrieves account and active session data, and sets initial balances of subscriber accounts. A subscriber account is a record of credit and debit entries in the database that track a subscriber's use of a particular network resource. You can also use the db-engine processor to dynamically adjust interim accounting intervals based on a service or based on a subscriber's remaining resources and use of the network for that service. Configuring the db-engine processor involves the following tasks:

- [Recording Balance Changes and Calculating the Average Usage Rate \(SRC CLI\) on page 69](#)
- [Configuring the Initial Balance and Status of a Subscriber Account in the External Database \(SRC CLI\) on page 70](#)
- [Configuring Scripts That Update Accounts \(SRC CLI\) on page 71](#)
- [Configuring the Interim Account Interval and Usage Metric of a Service in the External Database \(SRC CLI\) on page 72](#)
- [Variables Used to Define the Interim Accounting Interval for Services on page 72](#)
- [Variables Used to Define the Usage Metric for Services on page 75](#)
- [Sample Formulas for Usage Metrics for the SRC VTA on page 76](#)

### Recording Balance Changes and Calculating the Average Usage Rate (SRC CLI)

You can configure the db-engine processor to record account balance changes and calculate the average rate at which the subscriber is consuming volume in units per second, called the `averageUsageRate`.

The **session-history-depth** option determines the number of historical service session records that are used to calculate the `averageUsageRate` for a service. When you configure the interim accounting interval, you can specify the variable `averageUsageRate_serviceName`. If this variable is specified, the VTA calculates the `averageUsageRate` for the specified service by querying the history sessions of the service over the most recent *X* hours, where *X* is defined by the **session-history-depth** option.

For more information, see “[Configuring the Interim Account Interval and Usage Metric of a Service in the External Database \(SRC CLI\)](#)” on page 72 and “[Variables Used to Define the Interim Accounting Interval for Services](#)” on page 72.

Use the following statements to configure the VTA to record balance changes and specify the session history depth:

```
shared vta group name processor db-engine {
```

```
record-balance-change;  
session-history-depth;  
}
```



**NOTE:** The CLI editing level must be set to expert to set the `session-history-depth` option.

To configure the VTA to record balance changes and specify the session history depth:

1. From configuration mode, access the statement used to configure the db-engine processor. For example, to configure the db-engine processor for the VTA group called `vta1`:

```
[edit]  
user@host# edit shared vta group vta1 processor db-engine
```

2. (Optional) Configure the VTA to record balance changes.

```
[edit shared vta group vta1 processor db-engine]  
user@host# set record-balance-change
```

This option is set or not set. If it is set, every account balance change action is recorded in the database. Not setting this option requires fewer database updates and less database space.

3. (Optional) Specify the session-history-depth (in hours) used to calculate the `averageUsageRate` for a service.

```
[edit shared vta group vta1 processor db-engine]  
user@host# set session-history-depth
```

## Configuring the Initial Balance and Status of a Subscriber Account in the External Database (SRC CLI)

Use the following statements to set the initial balance and status of a subscriber account:

```
shared vta group name processor db-engine account account{  
  initial-balance initial-balance;  
  initial-status initial-status;  
}
```

To set the initial balance and status of a subscriber account:

1. From configuration mode, access the statement used to configure database accounts. For example, to configure an account called `BoughtQuota` in the VTA group called `vta1`:

```
[edit]  
user@host# edit shared vta group vta1 processor db-engine account BoughtQuota
```

2. Specify the initial balance for the account.

```
[edit shared vta group vta1 processor db-engine account BoughtQuota]
```

```
user@host# set initial-balance initial-balance
```

Enter the value as an integer in the range -9223372036854775807 through 9223372036854775807.

3. Specify the initial status for the account.

```
[edit shared vta group vta1 processor db-engine account BoughtQuota]
user@host# set initial-status initial-status
```

Enter the initial status as a text string—for example “active.”

## Configuring Scripts That Update Accounts (SRC CLI)

You can set up scripts to update balances in the accounts from which the usage of a service is charged and update accounts by assigning values to variables for the account balances.

Use the following statements to configure scripts that update accounts:

```
shared vta group name processor db-engine account-update-script name{
  script script;
}
```

To configure scripts that update accounts:

1. From configuration mode, access the statement used to configure scripts that update accounts. For example, to configure a script called DebtQuotaUsage in the VTA group called vta1:

```
[edit]
user@host# edit shared vta group vta1 processor db-engine account-update-script
DebtQuotaUsage
```

2. Specify the script parameters.

```
[edit shared vta group vta1 processor db-engine account-update-script
DebtQuotaUsage]
user@host# set script script
```

Enter a JavaScript program that updates a subscriber's account. The script can refer to the name of any attributes in the event being processed. An account can be updated by assigning values to the following parameters:

- `balance_<accountName>`—Values written to this parameter are put in the balance field of the account.
- `status_<accountName>`—Values written to this parameter are put in the status field of the account.
- `lastUpdateTime_<accountName>`—Values written to this parameter are put in the `last_update_time` field of the account.

For example:

```
[edit shared vta group vta1 processor db-engine account-update-script
DebtQuotaUsage]
user@host# set script <balance_PeriodicQuota>=<balance_PeriodicQuota>-\\
<currentUsage>;<lastUpdateTime_PeriodicQuota>=<currentTime>;
```

## Configuring the Interim Account Interval and Usage Metric of a Service in the External Database (SRC CLI)

To configure how the db-engine processor manages services, you need to specify the usage metric and the interim accounting interval options. Both of these options are configured with JavaScript variables.

Use the following statement to configure how the VTA manages services:

```
shared vta group name processor db-engine service name {
  interim-interval-function interim-interval-function;
  usage-metric usage-metric;
}
```



**NOTE:** The CLI editing level must be set to expert for this task.

To configure how the db-engine processor manages services:

1. From configuration mode, access the statement that configures the db-engine processor to manage services and specify the name of the service. For example, to configure a service called QuotaLocal in the VTA group vta1:

```
[edit]
user@host# edit shared vta group vta1 processor db-engine service QuotaLocal
```

2. Specify the JavaScript function that defines the usage metric for the service.

```
[edit shared vta group vta1 processor db-engine service QuotaLocal]
user@host# set usage-metric usage-metric
```

3. Specify the JavaScript variables that define the interim accounting interval for the service.

```
[edit shared vta group vta1 processor db-engine service QuotaLocal]
user@host# set interim-interval-function interim-interval-function
```

## Variables Used to Define the Interim Accounting Interval for Services

### Current Service Variables

---

Use the variables described in this section to define a formula for the interim accounting interval.

*lastInterimTime*



- Last interim time interval.
- Value—Number of seconds in the range 1–2147483647

#### ***sessionLength***

- Length of the current session.
- Value—Number of seconds in the range 0–2147483647; value is 0 when the SRC VTA is calculating the interim time of start events. For other events, value is set by the PA\_SESSION\_TIME attribute.

#### ***maxUsageRate***

- Maximum rate at which the subscriber can use network resources according to the formula described in [Table 12 on page 75](#).
- Value—Integer in the range 0–9223372036854775807
- Guidelines—This formula corresponds to the usage formula for the same service as the interim formula.

The maxUsageRate variable is calculated for a service by means of the following values for the variables in the corresponding usage formula:

- upStreamBytes=PA\_UPSTREAM\_BANDWIDTH
  - downStreamBytes=PA\_DOWNSTREAM\_BANDWIDTH
  - interimTime=lastInterimTime
  - upStreamPackets=0
  - downStreamPackets=0

If you use the parameters upStreamPackets (PA\_IN\_PACKETS) and downStreamPackets (PA\_OUT\_PACKETS) in the usage formula and at the same time maxUsageRate in the interim interval formula, the maxUsageRate is not accurate, because the values for maximum upStreamPackets and downStreamPackets are unknown.

#### ***averageUsageRate***

- Average rate at which the subscriber is consuming volume in units per second. The unit can be a value such as dollars, bytes, or packets. The type of unit depends on the value specified in the formula. Measurement begins when the service starts.
- Value—Integer in the range 0–9223372036854775807; the value is 0 when the SRC VTA is calculating the interim time of start events.

For other events, the value is the usage formula divided by PA\_SESSION\_TIME. The usage formula is calculated from PA\_IN\_PACKETS, PA\_OUT\_PACKETS, PA\_OUT\_OCTETS, PA\_IN\_OCTETS, and PA\_SESSION\_TIME.

#### ***latestUsageRate***

- Rate of service usage since the last usage report.
- Value—Integer in the range 0–9223372036854775807; the value is 0 when the SRC VTA is calculating the interim time of start events.

The value is calculated by using the result of the usage formula divided by the length of the service session since the previous usage report for the same service.

### Other Service Variables

---

Use the variables described in this section to define an interim accounting formula that depends on usage of another service tracked by the VTA.

System requirements to calculate service usage, in the form of the `averageUsageRate` and the `sessionLength` variables, can affect system performance. Using a longer interim interval means that there are fewer interim events to process, which requires fewer system resources.

#### ***averageUsageRate\_<serviceName>***

- Average rate at which the service is consuming volume in units per second. The unit can be a value such as dollars, bytes, or packets. The type of unit depends on the value specified in the interim accounting formula. Measurement begins when the service starts.
- Value—Integer in the range 0–9223372036854775807; the value is 0 when the SRC VTA is calculating the interim time of start events.
- Guidelines—Service names can contain alphanumeric characters and dashes (–).

#### ***sessionLength\_<serviceName>***

- Length of a service session for the service.
- Value—Integer in the range 0–2147483647; the value is 0 when the SRC VTA is calculating the interim time of start events.
- Guidelines—Service names can contain alphanumeric characters and dashes (–).

### Account Balance Variable

---

Use the variable described in this section to obtain balance information from each of the subscriber's accounts.

#### ***balance\_<accountName>***

- Balance for the specified account before the new usage value is applied.
- Value—Integer in the range 0–9223372036854775807
- Example—balance\_PeriodicQuota refers to the balance for the PeriodicQuota account.

### Sample Formulas for Interim Accounting Interval

Table 12 on page 75 provides examples of formulas to dynamically adjust the interim accounting interval for a service.

**Table 12: Examples of Interim Accounting Interval**

Formula	Description
return 900	Accounting interval is fixed at 900 seconds (15 minutes).
return (<balance_Periodic> + <balance_Bought>) /<maxUsageRate>	Minimum time required for the subscriber to empty the periodic and bought accounts.
return <sessionLength> >= 60*15 ? (<balance_Periodic> + <balance_Bought>) /<averageUsageRate>/2 : (<balance_Periodic> + <balance_Bought>) /<maxUsageRate>	Half the time required for the subscriber to empty the accounts at the current average rate, or the minimum time if the session is shorter than 15 minutes.  Because the average rate may not be representative early in the session, check when the account is half empty.

### Variables Used to Define the Usage Metric for Services

A usage metric is a formula that calculates usage based on an accounting event for the specified service. The formula is specified in the form of a JavaScript program and it can specify variables. When you configure a service, you need to specify the interim interval and the usage metric.

For the usage metric, you define a formula that determines the use of network resources for a service. Each service in a VTA can use a different formula. You can configure the SRC VTA software to evaluate this formula for every accounting event it receives from the SAE for each quota service. The VTA can then debit the result from the accounts.

Use the variables described in this section to define the formula. See “[Sample Formulas for Usage Metrics for the SRC VTA](#)” on page 76 for examples of usage metric formulas.

#### *downStreamBytes*

- Amount of data that the subscriber downloaded from the network since the last accounting event.
- Value—Number of bytes in the range 0–9223372036854775807

#### *downStreamPackets*

- Number of data packets that the subscriber downloaded from the network since the last accounting event.
- Value—Integer in the range 0–9223372036854775807
- Guidelines—Do not use `downStreamPackets` in a usage formula and `maxUsageRate` in the interim interval formula for the same service at the same time.

***interimTime***

- Time since the last accounting event.
- Value—Number of seconds in the range 0–2147483647
- Guidelines—Generally, this value equals the interim accounting interval; however, it may exceed the interim accounting interval if an accounting event is lost. Similarly, the value may be less than the interim accounting interval if a stop event occurs in the middle of an accounting interval.

***upStreamBytes***

- Amount of data that the subscriber uploaded to the network since the last accounting event.
- Value—Number of bytes in the range 0–9223372036854775807

***upStreamPackets***

- Number of data packets that the subscriber uploaded to the network since the last accounting event.
- Value—Integer in the range 0–9223372036854775807
- Guidelines—Do not use `upStreamPackets` in a usage formula and `maxUsageRate` in the interim interval formula for the same service at the same time.

**Sample Formulas for Usage Metrics for the SRC VTA**

Table 13 on page 76 provides examples of usage formulas.

**Table 13: Examples of Formulas That Calculate Use of Network Resources**

Formula	Description	Function
<code>return &lt;upStreamBytes&gt; + &lt;downStreamBytes&gt;</code>	Number of bytes sent and received by the subscriber.	Tracks volume of data that the subscriber transfers.
<code>return 2*&lt;upStreamBytes&gt; + &lt;downStreamBytes&gt;</code>	Twice the number of sent bytes plus the number of received bytes.	Allows higher charges for subscribers who are operating servers.

**Table 13: Examples of Formulas That Calculate Use of Network Resources**  
(continued)

Formula	Description	Function
return <interimTime>	Time the subscriber is connected.	Tracks time that the subscriber connects rather than volume of data transfer.
return <downStreamBytes>/<interimTime>	Rate of downstream data transfer.	Allows higher charges for higher transfer rates.
QuotaInternet formula:  return <upStreamBytes> + <downStreamBytes>– (<upStreamPackets> + <downStreamPackets>)*20  QuotaLocal formula:  return (<upStreamBytes> + <downStreamBytes>– (<upStreamPackets> + <downStreamPackets>)*20)/2	Formulas for separate, complementary services in a single VTA.  The following expression returns the total number of bytes in the IP headers of packets uploaded and downloaded by the service, and as such is not subscriber data. It is not counted as usage.  (<upStreamPackets>+ <downStreamPackets>)*20	Provides support for two services: QuotaInternet for Internet service and QuotaLocal for local service.  Allows higher charges for Internet service than for local service. By allocating a fixed usage limit for both services to each subscriber, the formula encourages subscribers to access local resources due to decreased cost.

**Related Documentation**

- [Overview of Adjusting the Interim Accounting Interval on page 31](#)
- [Overview of Managing VTA Accounts and Sessions on page 30](#)
- 

**Configuring the Mailer Processor for the SRC VTA Group (SRC CLI)**

Configuring the mailer processor involves the following tasks:

- [Configuring the SRC VTA Mailer Processor to Send E-Mail Notifications \(SRC CLI\) on page 77](#)
- [Configuring the SRC VTA to Send E-Mail Notifications \(SRC CLI\) on page 78](#)

**Configuring the SRC VTA Mailer Processor to Send E-Mail Notifications (SRC CLI)**

Use the SRC VTA mailer processor to specify the SMTP server to use for e-mail messages that the SRC VTA sends to subscribers.

Use the following statement to configure the mailer processor:

```
shared vta group name processor mailer {  
    smtp-server smtp-server;  
    smtp-server-password smtp-server-password;  
    smtp-server-port smtp-server-port ;  
    smtp-server-user smtp-server-user;  
}
```

1. From configuration mode, access the statement that configures the VTA mailer processor. For example, to configure the mailer processor for the VTA group called `vta1`:

```
[edit]  
user@host# edit shared vta group vta1 processor mailer
```

2. Configure the SMTP server used for outgoing e-mail by specifying the host name or IP address of the SMTP server. For example, set the SMTP server to a host called `mail.example.com`:

```
[edit shared vta group vta1 processor mailer]  
user@host# set smtp-server mail.example.com
```

3. Configure the password used to access the SMTP server.

```
[edit shared vta group vta1 processor mailer]  
user@host# set smtp-server-password smtp-server-password
```

4. Configure the port used to connect to the SMTP server.

```
[edit shared vta group vta1 processor mailer]  
user@host# set smtp-server-port smtp-server-port
```

5. Configure the username for accessing the SMTP server. `mail.example.com`:

```
[edit shared vta group vta1 processor mailer]  
user@host# set smtp-server-user smtp-server-user
```

6. Commit your configuration.

```
[edit shared vta group vta1 processor mailer]  
user@host# commit
```

## Configuring the SRC VTA to Send E-Mail Notifications (SRC CLI)

To configure the SRC VTA to send e-mail notifications, you need to configure an event handler that specifies an action that calls the *mailer-send* function.

1. From configuration mode, access the statement that configures an action for the VTA event handler. For example, to create an action called `act1` for the VTA group called `vta1`:

```
[edit]  
user@host# edit shared vta group vta1 action act1
```

2. Configure the action to call the *mailer-send* function.

```
[edit shared vta group vta1 action act1]  
user@host# set function mailer-send
```

3. Configure the parameters for the mailer-send function.

```
[edit shared vta group vta1 action act1]  
user@host# set parameter
```

Specify the following parameters for the mailer-send function:

- Recipient—Address of the e-mail recipient
  - From—Address of the e-mail sender
  - Subject—Subject of the e-mail
  - Text—Text of the e-mail
4. Configure what to do if an error occurs. For example, to configure the event handler to go to the next action if an error occurs:

```
[edit shared vta group vta1 action act1]  
user@host# set on-error go-to-next-action
```

5. Configure the event handler and specify the action you configured in the previous steps of this procedure.

See [“Configuring Event Handlers \(SRC CLI\)” on page 66](#).

---

## Configuring the SRC VTA Scripts Processor (SRC CLI)

---

The scripts processor can invoke external executable scripts or JavaScript programs. We recommend using JavaScript programs for better performance.

- External scripts are executable programs, such as shell scripts, that are available on the VTA's host. Each external script can perform a task and return a value. If the script returns a value, the value can be added to the current event as an event attribute.
- JavaScript programs are used to process attributes of a VTA event. For example, a JavaScript program can convert a VTA event attribute in a timestamp to a date string and add it to the event as a new attribute. The attribute can then be used for subsequent actions, such as sending an e-mail notification to the subscriber. The JavaScript program can refer to any attributes of the event being processed and it can return a value.

Configuring the SRC VTA scripts processor involves the following tasks:

- [Configuring the SRC VTA to Run Scripts on page 80](#)
- [Configuring JavaScript Programs on page 80](#)
- [Configuring External Scripts on page 81](#)

## Configuring the SRC VTA to Run Scripts

To configure the scripts processor:

1. Configure a JavaScript program or an external script.
  - See [“Configuring JavaScript Programs” on page 80](#)
  - See [“Configuring External Scripts” on page 81](#)
2. Configure an action for the event handler that calls the appropriate script function.

The action specifies a function that the scripts processor performs on events. You can set up an action to run either an external script or a JavaScript program.

- Specify the *scripts-run-javascript* function if the script is a JavaScript.
- Specify the *scripts-run-external-script* function if the script is an external script.

When you specify these functions, you need to specify the name of a script you previously configured by using the **shared vta group id processor scripts** statement. If the script configuration contains a return-attribute and return-type, an attribute with that name and type are added to the event after the script is executed.

See [“Configuring Actions for a VTA Event Handler \(SRC CLI\)” on page 65](#)

3. Configure an event handler and specify the action you configured in Step 2.

See [“Configuring Event Handlers \(SRC CLI\)” on page 66](#)

## Configuring JavaScript Programs

Use the following statement to configure a JavaScript program for the SRC VTA:

```
shared vta group name processor scripts javascript name {  
  script script;  
  return-type return-type;  
  return-attribute return-attribute;  
}
```

1. From configuration mode, access the statement that configures a JavaScript script and specify the name of the JavaScript program. For example, to configure a JavaScript script called js1:

```
[edit]  
user@host# edit shared vta group vta1 processor scripts javascript js1
```

2. Specify a function body in JavaScript.

```
[edit shared vta group vta1 processor scripts javascript js1]  
user@host# set script script
```

To refer to the event attributes being processed, include the attribute name delimited by angle brackets (<and>). The JavaScript program can verify whether the event has the referenced attribute. If a referenced attribute does not exist in the event, the attribute's value is null. The JavaScript program must return a value.



3. (Optional) Specify the Java type of the return attribute.

```
[edit shared vta group vta1 processor scripts javascript js1]
user@host# set return-type return-type
```

Where the **return-type** is one of the following:

- Integer
- Long
- Float
- Double
- String
- Boolean

4. (Optional) Specify the name of the return attribute.

```
[edit shared vta group vta1 processor scripts javascript js1]
user@host# set return-attribute return-attribute
```

Specify an attribute that provides the return value of the script as a valid Java identifier that subsequent actions and event handlers can refer to. If this attribute is not set, the return value is not added to the event as an event attribute.

JavaScript scripts must return an attribute value. The name of a return attribute cannot start with an underscore (\_) because these event attributes are reserved for internal use.

## Configuring External Scripts

Use the following statement to configure an external script for the SRC VTA:

```
shared vta group name processor scripts external-script name {
  full-path full-path;
  parameters parameters;
  return-type return-type;
  return-attribute return-attribute;
}
```

1. From configuration mode, access the statement that configures an external script and specify the name of the external script. For example, to configure an external script called script1:

```
[edit]
user@host# edit shared vta group vta1 processor scripts external-script script1
```

2. Specify the full path name to the external script.

```
[edit shared vta group vta1 processor scripts external-script script1]
user@host# set full-path full-path
```

You must specify a fully qualified path to an executable program on the local file system.

3. Specify the parameters required by the script as [a b c]. When an action invokes the script, the action must supply values for these parameters in the same order as they are defined here.

```
[edit shared vta group vta1 processor scripts external-script script1]  
user@host# set parameters parameters
```

4. (Optional) Specify the Java type of the return attribute.

```
[edit shared vta group vta1 processor scripts external-script script1]  
user@host# set return-type return-type
```

Where return-type is one of the following:

- Integer
- Long
- Float
- Double
- String
- Boolean

5. (Optional) Specify the name of the return attribute.

```
[edit shared vta group vta1 processor scripts external-script script1]  
user@host# set return-attribute return-attribute
```

Specify an attribute that provides the return value of the script as a valid Java identifier that subsequent actions and event handlers can refer to. If this attribute is not set, the return value is not used by the event. The external script returns the value by printing to standard output.



**NOTE:** The name of a return attribute cannot start with an underscore (\_) because these event attributes are reserved for internal use.

---

**Related  
Documentation**

- [Configuring Scripts That Update Accounts \(SRC CLI\) on page 71](#)
- [Configuring the SRC VTA to Run Scripts on page 80](#)

## Configuring VTA Logging (SRC CLI)

The SRC VTA generates messages that you can save in logs—either by writing the messages to text files or by using the system log facilities. Configuring VTA logging involves the following tasks:

- [Logging Events Messages to a Text File on page 83](#)
- [Logging Events Messages to a System Logging Server on page 86](#)

### Logging Events Messages to a Text File

Use this procedure to configure the SRC VTA to save event messages in text files.

Use the following statement to configure the SRC VTA to save event messages in text files:

```
shared vta group name logger name file {
  filename filename;
  filter filter;
  maximum-file-size maximum-file-size;
  rollover-filename rollover-filename;
}
```

1. From configuration mode, access the statement that configures logging to a text file. For example, to configure a logger called `vta1-logger` for the VTA group called `vta1`:

```
[edit]
user@host# edit shared vta group vta1 logger vta1-logger file
```

2. Specify the path and filename of the current log file. For example:

```
[edit shared vta group vta1 logger vta1-logger file]
user@host# set filename pathname/filename.log
```

Make sure you have write access to the folder.

3. (Optional) Specify a filter that determines the type of messages that this log file contains.

```
[edit shared vta group vta1 logger vta1-logger file]
user@host# set filter filter
```

The filter is specified in an expression. The software filters events by evaluating each subexpression from left to right. When the software finds a match, it logs or ignores the message accordingly. You can specify an unlimited number of subexpressions. The order in which you specify the subexpressions affects the result. Expressions have the format:

```
singlematch [,singlematch]
```

Where

```
singlematch—[!] ( <category> | ([<category>]/[<severity>] |
[<minimumSeverity>]-[<maximumSeverity>] ))
```

- `!`—Do not log matching events.
- `<category>`—SRC component that generated the event message. To log only events in a specific category, you can define the category, which is a text string that matches the name of a category. The text string is not case sensitive. For the names of categories, view a log file for a default filter. Juniper Networks Technical Assistance Center (JTAC) can also provide category names.
- `[<severity>] | [<minimumSeverity>]-[<maximumSeverity>]`—Name or number in the range 1–127. A higher number indicates a higher severity level. [Table 14 on page 84](#) shows common severity levels that you can specify by name.

**Table 14: Named Severity Levels**

Name	Severity Level
logmin	1
debug	10
info	20
notice	30
warning	40
error	50
crit	60
alert	70
emerg	80
panic	90
logmax	127

Enabling debug log messages has a negative effect on system performance. Do not enable debug log messages unless JTAC instructs you to do so.

You can define a severity level as follows:

- Specify an explicit severity. For example:  
warning—Defines only warning messages
- Specify a minimum severity and a maximum severity. For example:  
info-warning—Defines messages of minimum severity level of info and a maximum severity level of warning

- Accept the default minimum (logmin) or maximum (logmax) severity by omitting the minimum or maximum severity. For example:  
 info—Defines messages of minimum severity level info and maximum severity level logmax  
 -warning—Defines messages of minimum severity level logmin and maximum severity level warning
- Specify no severity to log all event messages.
- [Table 15 on page 85](#) shows some examples of filters.

**Table 15: Examples of Filters for Event Messages**

Syntax	Event Messages Saved
/	All event messages
/info-	Event messages of level info and higher from all categories
vta/debug	Debug events from the VTA category only
!vta,/debug	All debug events except those from the VTA category
!VtaMsg/info-,vtaMsg,vta	All messages from the VTA category, except those from the VtaMsg category with level lower than info

4. Specify the maximum file size. This option disables or enables and sets the maximum size of the log file and the rollover file.

```
[edit shared vta group vta1 logger vta1-logger file]
user@host# set maximum-file-size maximum-file-size
```

This is specified in number of kilobytes in the range 0–4294967295. The default is 1000000.

Do not set the maximum file size to a value greater than the available disk space.

5. Specify the rollover filename.

```
[edit shared vta group vta1 logger vta1-logger file]
user@host# set rollover filename rollover filename
```

Specify the path and filename of the rollover log file. When the log file reaches the maximum size, the software closes the log file and renames it with the name you specify for the rollover file. If a previous rollover file exists, the software overwrites it. The software then reopens the log file and continues to save event messages in it.

For example:

```
[edit shared vta group vta1 logger vta1-logger file]
user@host# set rollover filename vta_debug.alt
```

## Logging Events Messages to a System Logging Server

Use this procedure to configure the SRC VTA to save event messages on a system logging server.

Use the following statement to configure the SRC VTA to save event messages on a system logging server:

```
shared vta group name logger name syslog {  
    filter filter;  
    host host;  
}
```

1. From configuration mode, access the statement that configures logging to the system log facility. For example, to configure a logger called vta1-logger for the VTA group called vta1:

```
[edit]  
user@host# edit shared vta group vta1 logger vta1-logger syslog
```

2. (Optional) Specify a filter that determines the type of messages that this log file contains.

```
[edit shared vta group vta1 logger vta1-logger syslog]  
user@host# set filter filter
```

The filter is specified in an expression. The software filters events by evaluating each subexpression from left to right. When the software finds a match, it logs or ignores the message accordingly. You can specify an unlimited number of subexpressions. The order in which you specify the subexpressions affects the result. Expressions have the format:

singlematch [,singlematch]

Where

singlematch—[!] ( <category> | ([<category>]/[<severity>] |  
[<minimumSeverity>]-[<maximumSeverity>] ))

- **!**—Do not log matching events.
- **<category>**—SRC component that generated the event message. To log only events in a specific category, you can define the category, which is a text string that matches the name of a category. The text string is not case sensitive. For the names of categories, view a log file for a default filter. Juniper Networks Technical Assistance Center (JTAC) can also provide category names.
- **[<severity>] | [<minimumSeverity>]-[<maximumSeverity>]**—Name or number in the range 1–127. A higher number indicates a higher severity level. [Table 16 on page 87](#) shows common severity levels that you can specify by name.

Table 16: Named Severity Levels

Name	Severity Level
logmin	1
debug	10
info	20
notice	30
warning	40
error	50
crit	60
alert	70
emerg	80
panic	90
logmax	127

Enabling debug log messages has a negative effect on system performance. Do not enable debug log messages unless JTAC instructs you to do so.

You can define a severity level as follows:

- Specify an explicit severity. For example:  
warning—Defines only warning messages
- Specify a minimum severity and a maximum severity. For example:  
info-warning—Defines messages of minimum severity level of info and a maximum severity level of warning
- Accept the default minimum (logmin) or maximum (logmax) severity by omitting the minimum or maximum severity. For example:  
info—Defines messages of minimum severity level info and maximum severity level logmax  
-warning—Defines messages of minimum severity level logmin and maximum severity level warning
- Specify no severity to log all event messages.
- Example—[Table 17 on page 88](#) shows some examples of filters.

Table 17: Examples of Filters for Event Messages

Syntax	Event Messages Saved
/	All event messages
/info-	Event messages of level info and higher from all categories
vta/debug	Debug events from the VTA category only
!vta,/debug	All debug events except those from the VTA category
!VtaMsg/info-,vtaMsg,vta	All messages from the VTA category, except those from the VtaMsg category with level lower than info

3. Specify the host information for the system log server.

```
[edit shared vta group vta1 logger vta1-logger syslog]
user@host# set host host
```

Specify an IP address or name of a host that collects event messages with a standard system logging daemon.

#### Related Documentation

- [Logging Events Messages to a System Logging Server on page 86](#)
- [Overview of the SRC VTA on page 9](#)
- [How the SRC VTA Works on page 12](#)
- [Logging Events Messages to a Text File on page 83](#)

## Enabling, Disabling, and Restarting the VTA (SRC CLI)

After you create and commit a VTA shared configuration, a new top-level SRC component called *vta-name* automatically appears in the SRC CLI. This component is visible when you execute the following SRC CLI operational commands:

- **show component**
- **enable component *vta-name***
- **disable component *vta-name***
- **restart component *vta-name***

Where the *vta-name* is the name you specified for the group name under the **edit shared vta group *name*** configuration statement. You can configure multiple VTA instances (groups), each with a unique name.

After the configuration for a VTA group is deleted and committed, the SRC component called *vta-name* automatically disappears from the SRC CLI.



When you issue the **show component** command, a component called VTA is always visible. However, this component cannot be enabled or disabled or restarted. It is only displayed so that you can view the version number of the VTA. Only VTA group components with names like *vta-name* can be enabled and disabled and restarted.

To enable a VTA:

- From operational mode, enable the VTA.  
`user@host> enable component vta-name`

To disable a VTA:

- From operational mode, enable the VTA.  
`user@host> disable component vta-name`

To restart a VTA:

- From operational mode, enable the VTA.  
`user@host> restart component vta-name`

- Related Documentation**
- [Overview of the SRC VTA on page 9](#)
  - [How the SRC VTA Works on page 12](#)

---

## Using One VTA Account for Multiple Subscriber Sessions

---

The SRC VTA allows multiple subscriber sessions to share the same VTA account. The SRC VTA debits usage for all the subscriber sessions from the account. When the account is empty, service sessions for all subscribers are stopped. When the account is refilled, the SRC VTA starts services for all subscriber sessions that share the account.

To use this feature, you use the `subscriberId` event attribute to map a group of subscribers to the VTA account. You then use the `accounting-id` attribute as the **subscriber-id-solution** parameter under the **shared vta group name** configuration statement. For information about this option, see [“Creating and Configuring a VTA Shared Group Configuration \(SRC CLI\)” on page 59](#). You also set up the NIC to use the accounting ID to look up the SAE that manages a subscriber.

To set up the SRC software to use one VTA account for multiple subscribers:

1. In the subscriber classifier script, assign a value to the `accountingUserId` attribute. For example, you could assign it to the `userName`, `interfaceName`, `loginName`, or a combination of classification criteria. The purpose of the assignment is to allow the SRC VTA to identify subscribers by many different subscriber attributes using `accountingUserId` as a wrapper.

For example, the following subscriber classifier script assigns the value of the `userName` to the `accountingUserId` attribute:

```
[<-retailerDn->?accountingUserId=<-userName->?sub?(uniqueID=<-userName->)]
```

2. Configure the SAE to publish the PA\_ACCOUNTING\_ID plug-in attribute in subscriber-tracking events to the NIC SAE agent plug-in.

See [Configuring Internal Plug-Ins \(SRC CLI\)](#).

3. Configure the NIC to use the OnePopAcctId NIC scenario.

See [Configuring the NIC \(SRC CLI\)](#).

4. Configure the VTA shared group configuration in the SRC CLI as follows:

- For the **subscriber-id-solution** option enter **accounting-id**.
- For the **nic-proxy** option, enter the name of the NIC proxy configuration you entered for the **shared vta nic-proxy-configuration name** statement. Be sure to configure the NIC proxy so that it uses a NIC that maps from the accounting ID to the SAE.

See ["Creating and Configuring a VTA Shared Group Configuration \(SRC CLI\)" on page 59](#).

5. (Optional) Set up an action to apply functions to all subscriber sessions that share the same VTA account. For example, the following action starts services for all subscriber sessions that have the same subscriber ID.

```
[edit shared vta group vta1]
user@host# edit action xyz
[edit shared vta group vta1 action xyz]
user@host# set function sae-start-service
[edit shared vta group pear action xyz]
user@host## show

function sae-start-service;
parameter;
```

Make sure that the **current-subscriber-only** option is set to false (not present). If it is set, delete it.

#### Related Documentation

- [Overview of the SRC VTA on page 9](#)
- [Managing VTA Accounts and Sessions on page 30](#)
- [Managing Subscriber Sessions and Service Sessions on page 30](#)
- [Configuring Subscribers and Subscriptions to VTA Services on page 43](#)

## CHAPTER 7

# Examples

- [Example: Limiting Subscriber Access Based on Account Balances on page 91](#)

### Example: Limiting Subscriber Access Based on Account Balances

---

The sample data provides an example called Quota that limits a subscriber's access rate based on the balances of accounts that record the subscriber's use of network resources. Subscribers receive a quota of transfer (upload and download) volume in two ways:

- Periodic quota—Volume that is periodically added to a subscriber's account. For example, a subscriber may receive a 25-MB periodic quota each month. The periodic quota is tracked in the periodic account.
- Bought quota—Additional volume that a subscriber can purchase and use at any time. For example, a subscriber may purchase 25 MB of bought quota in January, and use the bought quota between January and March. Bought quota is tracked in a bought account.

As a subscriber consumes volume, the SRC VTA debits the accounts, using first the periodic quota and, if no periodic quota is available, the bought quota.

Subscribers managed by a VTA require a subscription to quota services—services for which a VTA monitors and manages usage. You must configure these subscriptions to be activated when the subscriber logs in. When a subscriber logs in to the SAE, the SAE tries to activate the quota services. However, if neither the periodic account nor the bought account has a positive balance, the SRC VTA deactivates the quota services, and the SAE applies to the subscriber either the default policy or another policy that implements a service with a lower bandwidth.

The units of the accounts depend on the formula that you define to determine the use of network resources. With this formula, the SRC VTA calculates the change to the accounts and the interim accounting interval.

The Quota configuration example provides a VTA that operates as follows:

1. When a service session for the quota service starts, the SAE sends a start event to the SRC VTA.
2. The SRC VTA starts a VTA session that has the same identifier as the service session and a qualifier of zero.

3. When the SRC VTA receives the first interim update from the SAE in a VTA session, it records a balance change that details the use of resources and the event time for the VTA session. When the SRC VTA receives interim updates in the VTA session, it updates the use of resources and the event time in the balance change recorded previously.
  - If the periodic account contains sufficient resources to cover the balance change, the SRC VTA changes only the balance of that account.
  - If the periodic account does not contain sufficient resources for the change, the SRC VTA records one balance change for the resources available in that account and records another balance change for the difference in the bought account. In this case, the SRC VTA records subsequent balance changes to the bought account.
  - If neither account has sufficient resources, the SRC VTA deactivates the quota service.
4. If the SAE session ends, the VTA session ends. When a new service session starts, Steps 1 to 3 recur. However, a service session may last for several VTA sessions. In this case, the SAE and SRC VTA continue the process described in Step 3.
5. When an administrator replenishes the periodic quota, the SRC VTA ends the VTA session, finalizes all balance changes for the session, and records a credit to the periodic account.
6. When the subscriber buys additional volume, the SRC VTA ends the VTA session, finalizes all balance changes for the session, and records a credit to the bought account.
7. When the SRC VTA next receives an interim update event from the SAE, it starts a new VTA session. The SRC VTA obtains the start time for the VTA session from the SAE event, and records debits to the accounts as described in Step 3.

The SRC VTA always ends the VTA session when an administrator replenishes periodic quota or the subscriber buys volume. However, a service session may last for several billing periods. In this case, when the SRC VTA starts a new VTA session, it continues to assign the SAE session identifier to the VTA session and increments the qualifier by one. Keeping the VTA session within a billing period allows the SRC VTA to finalize balance changes.

**Related Documentation**

- [Loading Sample Data into a Juniper Networks Database \(SRC CLI\)](#)
- [Overview of the SRC VTA on page 9](#)
- [Identifying Subscribers, SAEs, and Sessions on page 30](#)
- [Managing VTA Accounts and Sessions on page 30](#)
- [Managing Subscriber Sessions and Service Sessions on page 30](#)

## PART 3

# Administration

- [Managing the SRC VTA on page 95](#)



## CHAPTER 8

# Managing the SRC VTA

- [Deleting Balance Change History Records from the Database \(SRC CLI\) on page 95](#)
- [Deleting Session History Records from the Database \(SRC CLI\) on page 95](#)
- [Deleting Subscriber VTA Accounts \(SRC CLI\) on page 96](#)
- [Modifying VTA Accounts and Service Sessions \(SRC CLI\) on page 96](#)
- [Terminating Sessions \(SRC CLI\) on page 97](#)
- [Viewing a Subscriber's VTA Accounts \(SRC CLI\) on page 98](#)
- [Viewing Balance Change History for a Subscriber \(SRC CLI\) on page 98](#)
- [Viewing a Subscriber's Session History \(SRC CLI\) on page 98](#)
- [Viewing VTA Performance Statistics \(SRC CLI\) on page 99](#)
- [Overview of Testing the VTA Configuration on page 102](#)
- [Testing VTA Events \(SRC CLI\) on page 105](#)

### Deleting Balance Change History Records from the Database (SRC CLI)

Use the **request vta group *vtaName* delete balance-changes before *date*** command to delete old balance change records from the VTA database. This command deletes balance change records from the VTA database that have a last updated time before the specified date.

- From operational mode, execute:

```
user@host> request vta group vtaName delete balance-changes before date
```

Specify the date in the format yyyy-mm-dd.

#### Related Documentation

- [Deleting Session History Records from the Database \(SRC CLI\) on page 95](#)
- [Deleting Subscriber VTA Accounts \(SRC CLI\) on page 96](#)

### Deleting Session History Records from the Database (SRC CLI)

Use the **request vta group *vtaName* delete sessions before *date*** command to delete old session history records from the VTA database. This command deletes session history records from the VTA database that have a last updated time before the specified date.

- From operational mode, execute:

```
user@host> request vta group vtaName delete sessions before date
```

Specify the date in the format yyyy-mm-dd.

**Related  
Documentation**

- [Deleting Balance Change History Records from the Database \(SRC CLI\) on page 95](#)
- [Deleting Subscriber VTA Accounts \(SRC CLI\) on page 96](#)

---

## Deleting Subscriber VTA Accounts (SRC CLI)

Use the **request vta group *vtaName* delete subscriber *subscriber-id*** command to delete a specified subscriber's VTA accounts and subscriber's sessions that have a status of "Closed." This command also deletes all balance changes (from administrative actions) and session balance changes (from session usage) associated with those accounts.

- From operational mode, execute:

```
user@host> request vta group vtaName delete subscriber subscriber-id
```

**Related  
Documentation**

- [Deleting Balance Change History Records from the Database \(SRC CLI\) on page 95](#)
- [Deleting Session History Records from the Database \(SRC CLI\) on page 95](#)

---

## Modifying VTA Accounts and Service Sessions (SRC CLI)

Use the **request vta group *vtaName* update-accounts *account-name* *account-name*** command to modify subscriber VTA accounts and service sessions. By using this command, you can modify either all service sessions in an account or a specific subscriber's service sessions in the account.

You can overwrite the status, description, last update time, or balance in a set of accounts. You can also change the existing account balance by a specific positive or negative amount.

- To modify all service sessions in an account, execute the following command from operational mode:

```
user@host> request vta group vtaName update-accounts account-name account-name
```

- To modify a specific subscriber's service sessions in the account, execute the following command from operational mode:

```
user@host> request vta group vtaName update-accounts account-name account-name  
subscriber-id subscriber-id
```

You can optionally add the **account-status** option if you only want to update accounts with a particular status.

Use one or more of the options described in [Table 18 on page 97](#) to modify the account.



Table 18: Arguments Used to Modify Accounts

Argument	Description
<b>new-status</b> <i>newAccountStatus</i>	Changes the status of the account or specified subscriber's service sessions.
<b>new-balance</b> <i>newAccountBalance</i>	Sets the account balance to the amount specified by the <i>newAccountBalance</i> . For example, an account starts with a balance of 10. You set the <i>newAccountBalance</i> to 2. The account now has a balance of 2.
<b>balance-change</b> <i>accountBalanceChangeAmount</i>	Adds the amount specified for the <i>accountBalanceChangeAmount</i> to the existing account balance. For example, an account starts with a balance of 10. You set the <i>accountBalanceChangeAmount</i> to 2. The account now has a balance of 12.
<b>balance-change-description</b> <i>description</i>	(Optional) Add a description for the balance change. If you specify this option, a new balance change record is created in the database that contains the specified description, as well as the amount by which the account balance was changed. If the account balance was not changed, this amount will be zero.
<b>terminate-sessions</b>	(Optional) Trigger a <b>callback:terminatesessions</b> event for each specified subscriber. The VTA ignores these events unless you configure an event-handler to process them.

When you execute either of these commands, the VTA takes the following steps:

- If you specify the **terminate-sessions** option, the VTA generates a **callback:terminatesessions** event for every subscriber that owns a selected account.
- The VTA updates all service sessions in the account, or all service sessions for the specified subscriber as requested. This is purely a database operation.
- For each account that is updated, the VTA creates an **account-update** event. This event contains the relevant account's old and new balances, and other information. The VTA ignores these events unless you configure an event handler to process this event type.

## Terminating Sessions (SRC CLI)

For better integration with billing systems, it is sometimes desirable to terminate a set of subscribers' VTA sessions at the end of a billing period.

- To terminate VTA sessions, from operational mode, execute:

```
user@host> request vta group vtaName terminate-sessions subscriber-id subscriber-id
```



**NOTE:** To use `request vta group vtaName terminate-sessions subscriber-id subscriber-id` to terminate sessions, you need to configure an event handler to process **callback:terminatesessions** events by invoking an action that in turn invokes the **db-engine-terminate-session** function. If you do not configure the event handler to process these events in this manner, executing this command does nothing.

## Viewing a Subscriber's VTA Accounts (SRC CLI)

---

- Purpose** View the list of all specified VTA accounts of a subscriber.
- Action** To view information about a subscriber's VTA account:
- ```
user@host> show vta accounts group name subscriber-id subscriberId account-name account-name
```
- The **group** and **subscriber-id** options are mandatory. The **account-name** option is optional.
- Related Documentation**
- [Viewing Balance Change History for a Subscriber \(SRC CLI\) on page 98](#)
  - [Viewing a Subscriber's Session History \(SRC CLI\) on page 98](#)

## Viewing Balance Change History for a Subscriber (SRC CLI)

---

- Purpose** View a list of the specified VTA accounts and all associated balance changes and session balance changes if the changes have a timestamp between the *from* and *to* dates. Balance changes (from administrative actions) and session balance changes (from service usage) are displayed in a single list and ordered by their timestamps from the oldest to most recent.
- Action** To view a list of the specified VTA accounts and all associated balance changes and session balance changes, from operational mode, execute:
- ```
user@host> show vta balance-changes group name subscriber-id subscriber-id account-name account-name from from to to
```
- Specify the date in the format yyyy-mm-dd.
- The system displays both balance changes from administrative actions and session balance changes from service usage. The from and to dates are optional. If you omit the to date, the system defaults to the infinite future. If you omit the from date, the system defaults to the previous day.
- Related Documentation**
- [Viewing a Subscriber's VTA Accounts \(SRC CLI\) on page 98](#)
  - [Viewing a Subscriber's Session History \(SRC CLI\) on page 98](#)

## Viewing a Subscriber's Session History (SRC CLI)

---

- Purpose** View a subscriber's VTA-tracked session history.
- Action** To view a list of the subscriber's VTA-tracked session history, from operational mode, execute:
- ```
user@host> show vta sessions group name subscriber-id subscriber-id from from to to
```
- Specify the date in the format yyyy-mm-dd.

The system displays a list of the subscriber's VTA-tracked sessions that have a last update time on or between the from and to dates. The list is ordered by session start time (not last update time), from the oldest to most recent.

The from and to dates are optional. If you omit the to date, the system defaults to the infinite future. If you omit the from date, balance changes are displayed for the last six days. Balance changes are displayed from the oldest to most recent.

- Related Documentation**
- [Viewing a Subscriber's VTA Accounts \(SRC CLI\) on page 98](#)
  - [Viewing Balance Change History for a Subscriber \(SRC CLI\) on page 98](#)

## Viewing VTA Performance Statistics (SRC CLI)

**Purpose** View performance statistics for a VTA's queue, event handlers, and actions.

**Action** To view performance statistics for a VTA group, from operational mode, execute:

```
user@host> show vta statistics performance group name
```

```
Uptime
Uptime (seconds) 44
Up Since          Tue Oct 18 14:53:04 EDT 2011

Event Queue
Configured max queue size          5000
Current queue size                  0
Events received                     0
Events rejected due to full queue   0
Events dispatched                   0
Events received in last 60s: Min time in SAE  0
Events received in last 60s: Avg time in SAE  0
Events received in last 60s: Max time in SAE  0
Events dispatched in last 60s: Min time in queue 0
Events dispatched in last 60s: Avg time in queue 0
Events dispatched in last 60s: Max time in queue 0
Events received/second in last 60s          0.0
Events dispatched/second in last 60s        0.0

Event Handler
Event handler name                  EndofBilling
Events received                     0
Events ignored                      0
Events processed                    0
Event processing failures            0
Successful processing in last 60s: Min time  0
Successful processing in last 60s: Avg time  0
Successful processing in last 60s: Max time  0
Successfully processed events/second in last 60s 0.0

Event Handler
Event handler name                  GetQuota
Events received                     0
Events ignored                      0
Events processed                    0
Event processing failures            0
Successful processing in last 60s: Min time  0
```

Successful processing in last 60s: Avg time 0  
Successful processing in last 60s: Max time 0  
Successfully processed events/second in last 60s 0.0

Event Handler  
Event handler name NoQuota  
Events received 0  
Events ignored 0  
Events processed 0  
Event processing failures 0  
Successful processing in last 60s: Min time 0  
Successful processing in last 60s: Avg time 0  
Successful processing in last 60s: Max time 0  
Successfully processed events/second in last 60s 0.0

Event Handler  
Event handler name QuotaRefilled  
Events received 0  
Events ignored 0  
Events processed 0  
Event processing failures 0  
Successful processing in last 60s: Min time 0  
Successful processing in last 60s: Avg time 0  
Successful processing in last 60s: Max time 0  
Successfully processed events/second in last 60s 0.0

Event Handler  
Event handler name RecordUsage  
Events received 0  
Events ignored 0  
Events processed 0  
Event processing failures 0  
Successful processing in last 60s: Min time 0  
Successful processing in last 60s: Avg time 0  
Successful processing in last 60s: Max time 0  
Successfully processed events/second in last 60s 0.0

Event Handler  
Event handler name SetInterim  
Events received 0  
Events ignored 0  
Events processed 0  
Event processing failures 0  
Successful processing in last 60s: Min time 0  
Successful processing in last 60s: Avg time 0  
Successful processing in last 60s: Max time 0  
Successfully processed events/second in last 60s 0.0

Action  
Action name CalcUsage  
Events received 0  
Events processed 0  
Event processing failures 0  
Successful processing in last 60s: Min time 0  
Successful processing in last 60s: Avg time 0  
Successful processing in last 60s: Max time 0  
Successfully processed events/second in last 60s 0.00

Action  
Action name CalculateInterim  
Events received 0

|                                                  |      |
|--------------------------------------------------|------|
| Events processed                                 | 0    |
| Event processing failures                        | 0    |
| Successful processing in last 60s: Min time      | 0    |
| Successful processing in last 60s: Avg time      | 0    |
| Successful processing in last 60s: Max time      | 0    |
| Successfully processed events/second in last 60s | 0.00 |

|                                                  |               |
|--------------------------------------------------|---------------|
| Action                                           |               |
| Action name                                      | DebitAccounts |
| Events received                                  | 0             |
| Events processed                                 | 0             |
| Event processing failures                        | 0             |
| Successful processing in last 60s: Min time      | 0             |
| Successful processing in last 60s: Avg time      | 0             |
| Successful processing in last 60s: Max time      | 0             |
| Successfully processed events/second in last 60s | 0.00          |

|                                                  |                    |
|--------------------------------------------------|--------------------|
| Action                                           |                    |
| Action name                                      | GetAccountBalances |
| Events received                                  | 0                  |
| Events processed                                 | 0                  |
| Event processing failures                        | 0                  |
| Successful processing in last 60s: Min time      | 0                  |
| Successful processing in last 60s: Avg time      | 0                  |
| Successful processing in last 60s: Max time      | 0                  |
| Successfully processed events/second in last 60s | 0.00               |

|                                                  |            |
|--------------------------------------------------|------------|
| Action                                           |            |
| Action name                                      | SetInterim |
| Events received                                  | 0          |
| Events processed                                 | 0          |
| Event processing failures                        | 0          |
| Successful processing in last 60s: Min time      | 0          |
| Successful processing in last 60s: Avg time      | 0          |
| Successful processing in last 60s: Max time      | 0          |
| Successfully processed events/second in last 60s | 0.00       |

|                                                  |                  |
|--------------------------------------------------|------------------|
| Action                                           |                  |
| Action name                                      | StartptspService |
| Events received                                  | 0                |
| Events processed                                 | 0                |
| Event processing failures                        | 0                |
| Successful processing in last 60s: Min time      | 0                |
| Successful processing in last 60s: Avg time      | 0                |
| Successful processing in last 60s: Max time      | 0                |
| Successfully processed events/second in last 60s | 0.00             |

|                                                  |                 |
|--------------------------------------------------|-----------------|
| Action                                           |                 |
| Action name                                      | StopptspService |
| Events received                                  | 0               |
| Events processed                                 | 0               |
| Event processing failures                        | 0               |
| Successful processing in last 60s: Min time      | 0               |
| Successful processing in last 60s: Avg time      | 0               |
| Successful processing in last 60s: Max time      | 0               |
| Successfully processed events/second in last 60s | 0.00            |

|                  |                  |
|------------------|------------------|
| Action           |                  |
| Action name      | TerminateSession |
| Events received  | 0                |
| Events processed | 0                |

```
Event processing failures                                0
Successful processing in last 60s: Min time             0
Successful processing in last 60s: Avg time             0
Successful processing in last 60s: Max time             0
Successfully processed events/second in last 60s 0.00
```

```
Action
Action name   test
Events received                                       0
Events processed                                       0
Event processing failures                             0
Successful processing in last 60s: Min time           0
Successful processing in last 60s: Avg time           0
Successful processing in last 60s: Max time           0
Successfully processed events/second in last 60s 0.00
```

```
user@host>
```

- Related Documentation**
- [Viewing a Subscriber's VTA Accounts \(SRC CLI\) on page 98](#)
  - [Viewing Balance Change History for a Subscriber \(SRC CLI\) on page 98](#)

---

## Overview of Testing the VTA Configuration

You can use VTA test commands to simulate events and test a specific VTA group configuration. You can simulate subscriber-tracking events, service-tracking events, and callback events.

There are two steps involved in configuring and testing a VTA configuration.

- First, you need to define the VTA test events. You define a test event by assigning values to a set of attributes. The attributes you can assign to the test event depends on the event type you specify. VTA test event configurations are stored in the Juniper Networks database so that you can reference them when executing VTA test commands from any C Series Controller.
- After you define a test event, you can reference the test event when you execute test commands.

[Table 19 on page 102](#) lists the event types and the configuration statements you use to define VTA test events.

**Table 19: VTA Event Types and Test Event Configuration Statements**

---

| Test Events | Configuration Statement Used to Define the Test Event | Event Type |
|-------------|-------------------------------------------------------|------------|
|-------------|-------------------------------------------------------|------------|

---

Table 19: VTA Event Types and Test Event Configuration Statements (*continued*)

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Subscriber-and service-tracking test events | <p>Use the following statement to configure subscriber- and service-tracking test events:</p> <pre>shared vta test-events event-name type type attributes {</pre> <p>The <i>event-name</i> variable is an arbitrary name you specify for the event. When you execute a test command, you specify this name for the <i>test-event</i> variable.</p> <p>See <a href="#">Table 20 on page 103</a> for setting the <i>attribute-name</i> variable for either subscriber- or service-tracking test events.</p>                                                           | <p>You can define the following event types for the <i>event-type</i> variable when defining subscriber-tracking events:</p> <ul style="list-style-type: none"> <li>• <b>user-start</b></li> <li>• <b>user-interim</b></li> <li>• <b>user-stop</b></li> </ul> <p>You can define the following event types for the <i>event-type</i> variable when defining service-tracking events:</p> <ul style="list-style-type: none"> <li>• <b>service-start:service-name</b></li> <li>• <b>service-interim:service-name</b></li> <li>• <b>service-stop:service-name</b></li> </ul> |
| Callback test events                        | <p>Use the following statement to configure callback test events:</p> <pre>shared vta test-events type call-back:name callback-attributes {</pre> <p>You can define the following attribute type for callback test events:</p> <ul style="list-style-type: none"> <li>• Boolean</li> <li>• Long</li> <li>• Integer</li> <li>• Double</li> <li>• Float</li> <li>• String</li> </ul> <p>The <i>event-name</i> variable is an arbitrary name you specify for the event. When you execute a test command, you specify this name for the <i>test-event</i> variable.</p> | <p>The only event type you can define for callback test events is <b>callback</b>.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

Table 20: Attributes for Subscriber- and Service-Tracking Test Events

| Attributes Supported for Subscriber-Tracking Events | Attributes Supported for Service-Tracking Events |
|-----------------------------------------------------|--------------------------------------------------|
| PA_ACCOUNTING_ID                                    | PA_ACCOUNTING_ID                                 |
| PA_AUTH_USER_ID                                     | PA_AGGR_ACCOUNTING_ID                            |
| PA_DHCP_PACKET                                      | PA_AGGR_AUTH_USER_ID                             |
| PA_EVENT_TIME                                       | PA_AGGR_LOGIN_NAME                               |
| PA_EVENT_TIME_MILLISECOND                           | PA_AGGR_SESSION_ID                               |
| PA_IF_RADIUS_CLASS                                  | PA_AGGR_USER_DN                                  |
| PA_IF_SESSION_ID                                    | PA_AGGR_USER_INET_ADDRESS                        |

Table 20: Attributes for Subscriber- and Service-Tracking Test Events (*continued*)

|                      |                           |
|----------------------|---------------------------|
| PA_INTERFACE_ALIAS   | PA_AUTH_USER_ID           |
| PA_INTERFACE_DESCR   | PA_DHCP_PACKET            |
| PA_INTERFACE_NAME    | PA_DOWNSTREAM_BANDWIDTH   |
| PA_LOGIN_ID          | PA_EVENT_TIME             |
| PA_LOGIN_NAME        | PA_EVENT_TIME_MILLISECOND |
| PA_NAS_INET_ADDRESS  | PA_IF_RADIUS_CLASS        |
| PA_NAS_IP            | PA_IF_SESSION_ID          |
| PA_NAS_PORT          | PA_IN_OCTETS              |
| PA_OPERATIONAL       | PA_IN_PACKETS             |
| PA_PORT_ID           | PA_INTERFACE_ALIAS        |
| PA_PRIMARY_USER_NAME | PA_INTERFACE_DESCR        |
| PA_RADIUS_CLASS      | PA_INTERFACE_NAME         |
| PA_ROUTER_NAME       | PA_LOGIN_ID               |
| PA_SESSION_TIMEOUT   | PA_LOGIN_NAME             |
| PA_SSP_HOST          | PA_NAS_INET_ADDRESS       |
| PA_SUBSCRIPTION_NAME | PA_NAS_IP                 |
| PA_TERMINATE_CAUSE   | PA_NAS_PORT               |
| PA_USER_DN           | PA_OPERATIONAL            |
| PA_USER_INET_ADDRESS | PA_OUT_OCTETS             |
| PA_USER_IP_ADDRESS   | PA_OUT_PACKETS            |
| PA_USER_MAC_ADDRESS  | PA_PORT_ID                |
| PA_USER_RADIUS_CLASS | PA_PRIMARY_USER_NAME      |
| PA_USER_TYPE         | PA_RADIUS_CLASS           |
|                      | PA_ROUTER_NAME            |
|                      | PA_SERVICE_NAME           |



Table 20: Attributes for Subscriber- and Service-Tracking Test Events (*continued*)

|                         |
|-------------------------|
| PA_SERVICE_SCOPE        |
| PA_SERVICE_SESSION_NAME |
| PA_SERVICE_SESSION_TAG  |
| PA_SESSION_ID           |
| PA_SESSION_TIME         |
| PA_SSP_HOST             |
| PA_SUBSCRIPTION_NAME    |
| PA_TERMINATE_CAUSE      |
| PA_UPSTREAM_BANDWIDTH   |
| PA_USER_DN              |
| PA_USER_INET_ADDRESS    |
| PA_USER_IP_ADDRESS      |
| PA_USER_MAC_ADDRESS     |
| PA_USER_RADIUS_CLASS    |
| PA_USER_TYPE            |

**Related Documentation** • [Overview of Configuring Event Handlers on page 17](#)

## Testing VTA Events (SRC CLI)

You can test VTA events for subscriber- and service-tracking events, as well as callback events.

- From operational mode, execute:

```
user@host> test vta events subscriber-id subscriber-identifier event-name event-name
group group
```

For the *event-name* variable, specify the event name you configured when you defined the test event with either the **shared vta test-events *name* callback-attributes *name*** or **shared vta test-events *name* attributes *name*** statement.

- Related Documentation**
- [Overview of Configuring Event Handlers on page 17](#)
  - [Overview of Testing the VTA Configuration on page 102](#)

## PART 4

# Troubleshooting

- [Troubleshooting Initial Configuration on page 109](#)



## CHAPTER 9

# Troubleshooting Initial Configuration

- [Troubleshooting Database Deadlocks on page 109](#)

### Troubleshooting Database Deadlocks

---

**Problem** The JBoss application server logs the following error when the database reports a deadlock—a condition in which the database operation cannot continue because two processes are both waiting for the other process to be completed before they proceed.

```
java.sql.SQLException: General error, message from server: "Deadlock found when trying
to get lock; Try restarting transaction"
```

**Solution** Deadlocks can occur for a variety of reasons in normal database operation. The SRC VTA resolves deadlocks in the database, and you should ignore this message.

- Related Documentation**
- [Installing Web Applications in the Application Server on page 57](#)
  - [Configuring a Database to Store Account and Session Data \(SRC CLI\) on page 41](#)
  - [Configuring the Initial Balance and Status of a Subscriber Account in the External Database \(SRC CLI\) on page 70](#)



## PART 5

# Index

- [Index on page 113](#)





# Index

## C

|                       |     |
|-----------------------|-----|
| conventions           |     |
| notice icons.....     | xii |
| text.....             | xii |
| customer support..... | xiv |
| contacting JTAC.....  | xiv |

## D

|                       |      |
|-----------------------|------|
| directory             |      |
| description.....      | 4    |
| directory server..... | 4    |
| documentation         |      |
| comments on.....      | xiii |

## L

LDAP (Lightweight Directory Access Protocol). See  
    directory; directory server

## M

|                  |      |
|------------------|------|
| manuals          |      |
| comments on..... | xiii |

## N

|                   |     |
|-------------------|-----|
| notice icons..... | xii |
|-------------------|-----|

## S

|                                           |    |
|-------------------------------------------|----|
| SRC components                            |    |
| description.....                          | 3  |
| SRC VTA (SRC Volume-Tracking Application) |    |
| accounts                                  |    |
| calculating interim interval.....         | 17 |
| calculating usage.....                    | 17 |
| description.....                          | 10 |
| getting balances.....                     | 17 |
| interim accounting interval, setting..... | 72 |
| managing with VTA Configuration           |    |
| Manager.....                              | 70 |
| service.....                              | 9  |

|                                            |            |
|--------------------------------------------|------------|
| subscriber.....                            | 9          |
| usage metric, setting.....                 | 69, 72, 75 |
| actions.....                               | 14         |
| configuring.....                           | 65         |
| overview.....                              | 21         |
| architecture.....                          | 11         |
| bought account.....                        | 10         |
| bought quota.....                          | 10         |
| example.....                               | 91         |
| connections to SRC components.....         | 11         |
| database engine processor                  |            |
| configuring.....                           | 70         |
| e-mail notifications, sending.....         | 77         |
| event attributes.....                      | 12, 17     |
| event handlers.....                        | 12, 17     |
| configuring.....                           | 66         |
| overview.....                              | 17         |
| event queue                                |            |
| configuring.....                           | 68         |
| events.....                                | 12, 17     |
| account update.....                        | 19         |
| callback.....                              | 20         |
| tracking events from SAE.....              | 19         |
| example                                    |            |
| limiting subscriber access.....            | 91         |
| function                                   |            |
| configuring.....                           | 65         |
| functions                                  |            |
| overview.....                              | 21         |
| how it works.....                          | 12         |
| initial account balance                    |            |
| setting .....                              | 70         |
| initial account status                     |            |
| setting .....                              | 70         |
| interval accounting interval, setting..... | 72         |
| JavaScript programs.....                   | 31         |
| logging events to a text file              |            |
| configuration.....                         | 83         |
| configuring.....                           | 83         |
| mail processor                             |            |
| configuring.....                           | 77         |
| on-error                                   |            |
| configuring.....                           | 65         |
| overview.....                              | 21         |
| operation process.....                     | 16         |
| overview.....                              | 9          |
| periodic account.....                      | 10         |
| periodic quota.....                        | 10         |
| example.....                               | 91         |

|                                                     |            |
|-----------------------------------------------------|------------|
| processors.....                                     | 12         |
| providing volume-based services.....                | 9          |
| quota service.....                                  | 10         |
| activation upon deposit.....                        | 45         |
| related configuration tasks                         |            |
| identifying subscribers, SAEs, and<br>sessions..... | 30         |
| NIC.....                                            | 45         |
| NIC proxies.....                                    | 46         |
| services and policies.....                          | 42         |
| subscribers and subscriptions.....                  | 43         |
| SAE events.....                                     | 19         |
| saving event messages on a system logging<br>server |            |
| configuration.....                                  | 86         |
| configuring.....                                    | 86         |
| script runner processor                             |            |
| configuring.....                                    | 80         |
| scripts                                             |            |
| external.....                                       | 80, 81     |
| JavaScript programs.....                            | 80         |
| services                                            |            |
| interim accounting interval, setting.....           | 72         |
| sessions.....                                       | 9          |
| closing.....                                        | 17         |
| subscriber login with IP address.....               | 45         |
| troubleshooting database deadlocks.....             | 109        |
| types.....                                          | 9          |
| usage metric, configuring.....                      | 69, 72, 75 |
| support, technical <i>See</i> technical support     |            |

## T

technical support  
    contacting JTAC.....xiv  
text conventions.....xii

## V

- Volume-Tracking Application See VTA
- Volume-Tracking Application (VTA)
  - account and session database connection, configuring See SRC CLI
  - database to store account and session data, configuring See SRC CLI
  - JDBC .jar file See installing
  - VTA group, configuring See SRC CLI

## VTA

|                                                              |        |
|--------------------------------------------------------------|--------|
| accounts and service sessions                                |        |
| modifying.....                                               | 96     |
| balance change history records                               |        |
| deleting.....                                                | 95     |
| performance statistics                                       |        |
| viewing.....                                                 | 99     |
| session history records                                      |        |
| deleting.....                                                | 95, 96 |
| sessions                                                     |        |
| terminating.....                                             | 97     |
| subscriber accounts                                          |        |
| viewing.....                                                 | 98     |
| subscriber balance changes                                   |        |
| viewing.....                                                 | 98     |
| subscriber session history                                   |        |
| viewing.....                                                 | 98     |
| testing configuration                                        |        |
| overview.....                                                | 102    |
| VTA events                                                   |        |
| testing.....                                                 | 105    |
| VTA (Volume-Tracking Application (VTA))                      |        |
| enabling .....                                               | 88     |
| VTA (Volume-Tracking Application)                            |        |
| account and session database connection,<br>configuration    |        |
| SRC CLI.....                                                 | 62     |
| database to store account and session data,<br>configuration |        |
| SRC CLI.....                                                 | 41     |
| VTA group, configuring                                       |        |
| SRC CLI.....                                                 | 59     |
| VTA Configuration Manager                                    |        |
| database accounts.....                                       | 70     |

## W

|                                         |    |
|-----------------------------------------|----|
| Web application server                  |    |
| application deployment.....             | 57 |
| channel stack                           |    |
| configuring.....                        | 52 |
| configuration statements                |    |
| SRC CLI.....                            | 50 |
| configuring the Web application server  |    |
| SRC CLI.....                            | 49 |
| installing Web applications inside..... | 57 |
| local properties                        |    |
| configuring .....                       | 51 |
| multicast-address                       |    |
| configuring.....                        | 52 |

|                           |    |
|---------------------------|----|
| node-id                   |    |
| configuring.....          | 53 |
| overview.....             | 32 |
| shared cluster name       |    |
| configuring.....          | 51 |
| shared cluster nodes      |    |
| configuring.....          | 53 |
| shared cluster properties |    |
| configuring.....          | 52 |
| starting.....             | 58 |

